2.1 ALGEBRAIC STRUCTURES

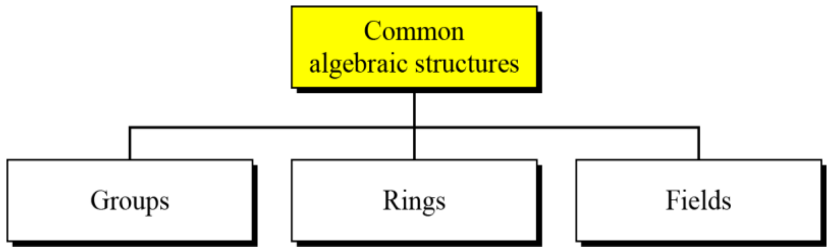


Figure 2.1 Common Algebraic Structures

2.1.1 Groups, Rings, Fields

Groups, rings, and fields are the fundamental elements of a branch of mathematics known as abstract algebra, or modern algebra.

Groups

A group G , sometimes denoted by {G,*} ,is a set of elements with a binary operation denoted by * that associates to each ordered pair (a,b) of elements G in an element(a*b) in , such that the following axioms are obeyed:

(A1) Closure: If a and b belong to G, then a*b is also in G. (A2) Associative: a*(b*c)=(a*b)*c for all a, b, , in G .
(A3) Identity element: There is an element e in G such that a*e=e*a=a for all in G .
(A4) Inverse element: For each a in G, there is an element a' in G such that a*a'=a'*a=e .

If a group has a finite number of elements, it is referred to as a **finite group**, and the **order** of the group is equal to the number of elements in the group. Otherwise, the group is an **infinite group**.

A group is said to be **abelian** if it satisfies the following additional condition:
(A5) Commutative: a*b = b*a for all a b, in G.

CYCLIC GROUP: A group is cyclic if every element of G is a power a^k (k is an integer) of a fixed element a£ G .The element is a said to generate the group G or to be a generator of
G.A cyclic group is always abelian and may be finite or infinite**.**

Rings

A ring R, sometimes denoted by {R, +, X}, is a set of elements with two binary operations, called addition and multiplication, such that for all a, b, c ,in R the following axioms are obeyed

(A1–A5) R is an abelian group with respect to addition; that is, R satisfies axioms A1 through A5. For the case of an additive group, we denote the identity element as 0 and the inverse of a as $-a$.

(M1) Closure under multiplication: If a and b belong to R, then ab is also in R.

(M2) Associativity of multiplication: $a(bc) = (ab)c$ for all a, b, c in R.

(M3) Distributive laws: $a(b + c) = ab + ac$ for all a, b, c in R.
$(a + b)c = ac + bc$ for all a, b, c in R.

A ring is said to be **commutative** if it satisfies the following additional condition:

(M4) Commutativity of multiplication: $ab = ba$ for all a, b in R.

Next, we define an integral domain, which is a commutative ring that obeys the following axioms

(M5) Multiplicative identity: There is an element 1 in R such that $a1 = 1a = a$ for all a in R.

(M6) No zero divisors: If a, b in R and $ab = 0$, then either $a = 0$ or $b = 0$.

Fields

A field F, sometimes denoted by {F, +, X}, is a set of elements with two binary operations, called addition and subtraction, such that for all a, b, c , in F the following axioms are obeyed

(A1–M6) F is an integral domain; that is, F satisfies axioms A1 through A5 and M1 through M6.

(M7) Multiplicative inverse: For each a in F, except 0, there is an element a^{-1} in F such that $aa^{-1} = (a^{-1})a = 1$.

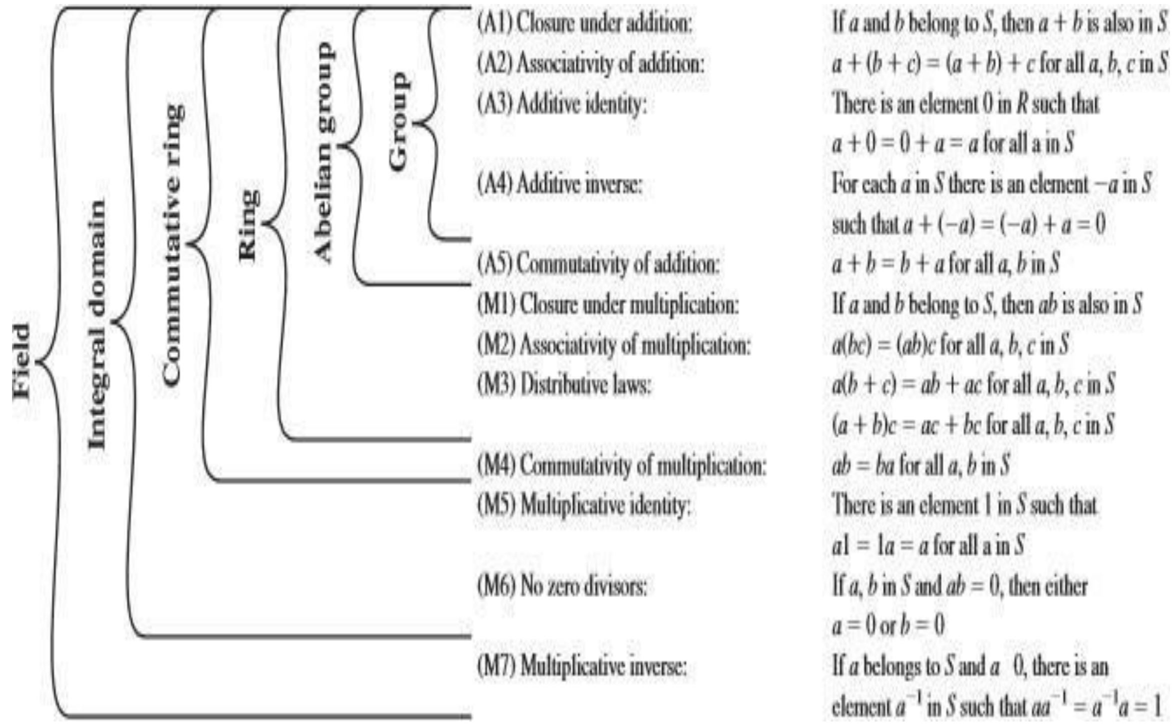


Figure 2.2 Groups, Ring and Field

2.2 MODULAR ARITHMETIC

If is an integer and n is a positive integer, we define a mod n to be the remainder when a is divided by n. The integer n is called the modulus. Thus, for any integer a, we can rewrite Equation as follows

$$a = qn + r \qquad 0 \leq r < n;\ q = \lfloor a/n \rfloor$$

$$a = \lfloor a/n \rfloor \times n + (a \bmod n)$$

$$\boxed{11 \bmod 7 = 4; \qquad -11 \bmod 7 = 3}$$

Two integers a and b are said to be **congruent modulo n**, if $(a \bmod n) = (b \bmod n)$. This is written as $a = b \pmod{n}$.[2]

$$\boxed{73 \equiv 4 \pmod{23}; \qquad 21 \equiv -9 \pmod{10}}$$

Note that if $a = 0 \pmod{n}$, then $n|a$.

Modular Arithmetic Operations

A kind of integer arithmetic that reduces all numbers to one of a fixed set [0,….,n-1] for some number n. Any integer outside this range is reduced to one in this range by taking the remainder after division by n.
Modular arithmetic exhibits the following properties

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
3. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

We demonstrate the first property. Define $(a \bmod n) = r_a$ and $(b \bmod n) = r_b$. Then we can write $a = r_a + jn$ for some integer j and $b = r_b + kn$ for some integer k. Then

$$\begin{aligned}(a + b) \bmod n &= (r_a + jn + r_b + kn) \bmod n \\ &= (r_a + r_b + (k + j)n) \bmod n \\ &= (r_a + r_b) \bmod n \\ &= [(a \bmod n) + (b \bmod n)] \bmod n\end{aligned}$$

The remaining properties are proven as easily. Here are examples of the three properties:

Table 2.1 Arithmetic Modulo 8

$11 \bmod 8 = 3; 15 \bmod 8 = 7$
$[(11 \bmod 8) + (15 \bmod 8)] \bmod 8 = 10 \bmod 8 = 2$
$(11 + 15) \bmod 8 = 26 \bmod 8 = 2$

$[(11 \bmod 8) - (15 \bmod 8)] \bmod 8 = -4 \bmod 8 = 4$
$(11 - 15) \bmod 8 = -4 \bmod 8 = 4$

$[(11 \bmod 8) \times (15 \bmod 8)] \bmod 8 = 21 \bmod 8 = 5$
$(11 \times 15) \bmod 8 = 165 \bmod 8 = 5$

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

(b) Multiplication modulo 8

w	$-w$	w^{-1}
0	0	—
1	7	1
2	6	—
3	5	3
4	4	—
5	3	5
6	2	—
7	1	7

(c) Additive and multiplicative inverses modulo 8

2.3 EUCLID' S ALGORITHM

One of the basic techniques of number theory is the Euclidean algorithm, which is a simple procedure for determining the greatest common divisor of two positive integers. First, we need a simple definition: Two integers are relatively prime if their only common positive integer factor is 1.

Greatest Common Divisor

Recall that nonzero b is defined to be a divisor ofa if a =mb for some m, where a,b, and m are integers. We will use the notation gcd(a , b) to mean the greatest common divisor of a and b .The greatest common divisor of a and b is the largest integer that divides both a and b

.We also define gcd(0,0) = 0.

Algorithm

The Euclid's algorithm (or Euclidean Algorithm) is a method for efficiently finding the greatest common divisor (GCD) of two numbers. The GCD of two integers X and Y is the largest number that divides both of X and Y (without leaving a remainder).

For every non-negative integer, a and any positive integer b

gcd (a, b) = gcd (b, a mod b)

```
Algorithm Euclids (a, b)
    α = a
    β = b
    while (β > 0)
        Rem = α mod β
        α = β
        β = Rem
    returnα
```

Steps for Another Method

$a = q_1 b + r_1; \; 0 < r_1 < b$

$b = q_2 r_1 + r_2; \; 0 < r_2 < r_1$

$r_1 = q_3 r_2 + r_3; \; 0 < r_3 < r_2$

$r_{n-2} = q_n r_{n-1} + r_n; \quad 0 < r_n < r_{n-1}$

$r_{n-1} = q_1 r_n + 0$

$d = \gcd(a, b) = r_n$

Example 1:

gcd (55, 22) = gcd (22, 55 mod 22)

= gcd (22, 11)

= gcd (11, 22 mod 11)

= gcd (11, 0)

gcd (55, 22) is 11

Example 2:

gcd (30, 50) = gcd (50, 30 mod 50)
= gcd (50, 30)
= gcd (30, 50 mod 30)
= gcd (30, 20)
= gcd (20, 30 mod 20)
= gcd (20, 10)
= gcd (10, 20 mod 10)
= gcd (10, 0)

gcd (30, 50) is 10

Another Method

To find gcd (30,50)

50	= 1 x 30 + 20	gcd (30, 20)
30	= 1 x 20 + 10	gcd (20,10)
20	= 1 x 10 + 10	gcd (10,10)
10	= 1 x 10 + 0	gcd (10,0)

Therefore, gcd (30,50) = 10

Example 3:

gcd (1970, 1066) = gcd (1066, 1970 mod 1066)
=gcd (1066, 904)
=gcd (904, 1066 mod 904)
=gcd (904, 162)
=gcd (162, 904 mod 162)
=gcd (162, 94)
=gcd (94, 162 mod 94)
=gcd (94, 68)
=gcd (68, 94 mod 68)
=gcd (68, 26)
=gcd (26, 68 mod 26)
=gcd (26, 16)
=gcd (16, 26 mod 16)
=gcd (16, 10)
=gcd (10, 16 mod 10)
=gcd (10, 6)
=gcd (6, 10 mod 6)
=gcd (6, 4)

=gcd (4, 6 mod 4)

=gcd (4, 2)

=gcd (2, 4 mod 2)

=gcd (2, 0)

gcd (1970, 1066) is 2

Another Method

To find gcd (1970, 1066)

1970	= 1 x 1066 + 904	gcd (1066, 904)
1066	= 1 x 904 + 162	gcd (904,162)
904	= 5 x 162 + 94	gcd (162, 94)
162	= 1 x 94 + 68	gcd (94, 68)
94	= 1 x 68 + 26	gcd (68, 26)
68	= 2 x 26 + 16	gcd (26, 16)
26	= 1 x 16 + 10	gcd (16, 10)
16	= 1 x 10 + 6	gcd (10, 6)
10	= 1 x 6 + 4	gcd (6, 4)
6	= 1 x 4 + 2	gcd (4, 2)
4	= 2 x 2 + 0	gcd (2, 0)

Therefore, gcd (1970, 1066) = 2

Extended Euclidean Algorithm

Extended Euclidean Algorithm is an efficient method of finding modular inverse of an integer.

Euclid's algorithm can be improved to give not just gcd (a, b), but also used to find the multiplicative inverse of a number with the modular value.

Example 1

Find the Multiplicative inverse of 17 mod 43

17-1 mod 43

17* X = mod 43

X= 17-1 mod 43

43 = 17 * 2 + 9

17 = 9 * 1 + 8

9 = 8* 1+ 1

Rewrite the above equation

9+8(-1) = 1 → (1)

17+9(-1) = 8 → (2)

43+17(-2) = 9 → (3)

Substitution

sub equ 2 in equ 1

(1)→ 9+8(-1) = 1 [Sub 17+9(-1) = 8]

9+(17+9(-1))(-1) = 1

9+17(-1)+9(1)=1

17(-1)+9(2) =1 → (4)

Now sub equ (3) in equ (4)

43+17(-2) = 9 → (3)

17(-1)+(43+17(-2))(2)=1

17(-1)+43(2)+17(-4)=1

17(-5)+43(2) = 1→(5)

Here -5 is the multiplicative inverse of 17. But inverse cannot be negative

17-1 mod 43 = -5 mod 43 = 38

So, 38 is the multiplicative inverse of 17.

Checking, 17* X ≡ 1 mod 43

17 * 38 ≡ 1 mod 43

646 ≡ 1 mod 43 (15*43 = 645)

Example 2

Find the Multiplicative inverse of 1635 mod 26

1635-1 mod 26

1635 = 26 (62) + 23

26 = 23 (1) + 3

23 = 3(7) + 2

3 = 2(1) + 1

Rewriting the above equation

3+2(-1) =1 → (1)

23+3(-7) = 2 → (2)

26+23(-1) = 3 → (3)

1635+26(-62) = 23 → (4)

Substitution

sub equ (2) in equ (1)

(2) => 23+3(-7) = 2

3+2(-1) =1

3+(23+3(-7))(-1) =1

3+23(-1)+3(7)=1

3(8)+23(-1) = 1 → (5)

sub equ (3) in equ (5)

26+23(-1) = 3 → (3)

(26+23(-1))(8) + 23 (-1) =1

26(8) + 23 (-8) + 23 (-1) =1

26 (8) + 23 (-9) =1 → (6)

Sub equ (4) in equ (6)

1635+26(-62) = 23 → (4)

26 (8) + (1635 + 26 (-62)) (-9) = 1

26 (8) + 1635 (-9) + 26 (558) =1

1635 (-9) + 26 (566) = 1 → (7)

From equ (7) -9 is inverse of 1635, but negative cannot be inverse.

1635-1 mod 26 = -9 mod 26 = 17

So, the inverse of 1635 is 17.

Checking, 1635* X ≡ 1 mod 26

1635 * 17 ≡ 1 mod 26

27795 ≡ 1 mod 26 (1069*26 = 27794)

2.4 CONGRUENCE AND MATRICES

Properties of Congruences

Congruences have the following properties:

1. $a \equiv b \pmod{n}$ if $n|(a - b)$.
2. $a \equiv b \pmod{n}$ implies $b \equiv a \pmod{n}$.
3. $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ imply $a \equiv c \pmod{n}$.

To demonstrate the first point, if $n|(a - b)$, then $(a - b) = kn$ for some k. So we can write $a = b + kn$. Therefore, $(a \bmod n) =$ (remainder when $b + kn$ is divided by n) $=$ (remainder when b is divided by n) $= (b \bmod n)$.

$23 \equiv 8 \pmod 5$	because	$23 - 8 = 15 = 5 \times 3$
$-11 \equiv 5 \pmod 8$	because	$-11 - 5 = -16 = 8 \times (-2)$
$81 \equiv 0 \pmod{27}$	because	$81 - 0 = 81 = 27 \times 3$

The remaining points are as easily proved.

Matrices

Matrix is a rectangular array in mathematics, arranged in rows and columns of numbers, symbols or expressions.

A matrix will be represented with their dimensions as l x m where l defines the row and m defines the columns

$$\begin{array}{cc} & m \text{ columns} \\ l \text{ rows} & \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{l1} & a_{l2} & \cdots & a_{lm} \end{bmatrix} \end{array}$$

Examples of Matrices
1. Row Matrix
2. Column Matrix
3. Square Matrix
4. Zero Matrixes
5. Identity Matrix

$$\underbrace{\begin{bmatrix} 2 & 1 & 5 & 11 \end{bmatrix}}_{\text{Row matrix}} \quad \underbrace{\begin{bmatrix} 2 \\ 4 \\ 12 \end{bmatrix}}_{\text{Column matrix}} \quad \underbrace{\begin{bmatrix} 23 & 14 & 56 \\ 12 & 21 & 18 \\ 10 & 8 & 31 \end{bmatrix}}_{\text{Square matrix}} \quad \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}_{\mathbf{0}} \quad \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{I}}$$

2.5 FINITE FIELDS

FINITE FIELDS OF THE FORM GF(p)

The finite field of order is generally written ; GF stands for Galois field,in honor of the mathematician who first studied finite fields

Finite Fields of Order p

For a given prime, , we define the finite field of order , , as the set of integers together with the arithmetic operations modulo **.**

The simplest finite field is GF(2). Its arithmetic operations are easily summarized:

+	0	1
0	0	1
1	1	0

Addition

×	0	1
0	0	0
1	0	1

Multiplication

w	$-w$	w^{-1}
0	0	—
1	1	1

Inverses

In this case, addition is equivalent to the exclusive-OR (XOR) operation, and multiplication is equivalent to the logical AND operation.

Finding the Multiplicative Inverse in It is easy to find the multiplicative inverse of an element in for small values of .You simply construct a multiplication table, such as shown in Table 2.2b,and the desired result can be read directly. However, for large values of ,this approach is not practical. p p GF(p) GF(p)

Table 2.2 Arithmetic in GF(7)

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

(a) Addition modulo 7

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

(b) Multiplication modulo 7

w	$-w$	w^{-1}
0	0	—
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

(c) Additive and multiplicative inverses modulo 7

2.5.1 Polynomial Arithmetic

We are concerned with polynomials in a single variable and we can distinguish three classes of polynomial arithmetic. • Ordinary polynomial arithmetic, using the basic rules of algebra. • Polynomial arithmetic in which the arithmetic on the coefficients is performed modulo

;that is,the coefficients are in .

Polynomial arithmetic in which the coefficients are in ,and the polynomials are defined modulo a polynomial whose highest power is some integer **.**

Ordinary Polynomial Arithmetic

A polynomial of degree (integer) is an expression of the form

A **polynomial** of degree n (integer $n \geq 0$) is an expression of the form

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 = \sum_{i=0}^{n} a_i x^i$$

where the a_i are elements of some designated set of numbers S, called the **coefficient set**, and $a_n \neq 0$. We say that such polynomials are defined over the coefficient set S.

A zero-degree polynomial is called a **constant polynomial** and is simply an element of the set of coefficients. An nth-degree polynomial is said to be a **monic polynomial** if $a_n = 1$.

In the context of abstract algebra, we are usually not interested in evaluating a polynomial for a particular value of x [e.g., $f(7)$]. To emphasize this point, the variable x is sometimes referred to as the **indeterminate**.

Addition and subtraction are performed by adding or subtracting corresponding coefficients. Thus, if

$$f(x) = \sum_{i=0}^{n} a_i x^i; \qquad g(x) = \sum_{i=0}^{m} b_i x^i; \qquad n \geq m$$

then addition is defined as

$$f(x) + g(x) = \sum_{i=0}^{m} (a_i + b_i)x^i + \sum_{i=m+1}^{n} a_i x^i$$

and multiplication is defined as

$$f(x) \times g(x) = \sum_{i=0}^{n+m} c_i x^i$$

where

$$c_k = a_0 b_k + a_1 b_{k-1} + \cdots + a_{k-1} b_1 + a_k b_0$$

As an example, let $f(x) = x^3 + x^2 + 2$ and $g(x) = x^2 - x + 1$, where S is the set of integers. Then

$$\begin{aligned} f(x) + g(x) &= x^3 + 2x^2 - x + 3 \\ f(x) - g(x) &= x^3 + x + 1 \\ f(x) \times g(x) &= x^5 + 3x^2 - 2x + 2 \end{aligned}$$

Polynomial Arithmetic with Coefficients in

Let us now consider polynomials in which the coefficients are elements of some field F; we refer to this as a polynomial over the field F. In that case, it is easy to show that the set of such polynomials is a ring, referred to as a polynomial ring. That is, if we consider each distinct polynomial to be an element of the set, then that set is a ring 8 when polynomial arithmetic is performed on polynomials over a field, then division is possible. Note that this does not mean that exact division is possible. Let us clarify this distinction. Within a field, given two elements and, the quotient is also an element of the field. However, given a ring that is not a field, in Ra /b ba Zp

$$\begin{array}{r}
x^3 + x^2 \qquad + 2 \\
+ \;\; (x^2 - x + 1) \\
\hline
x^3 + 2x^2 - x + 3
\end{array}$$

(a) Addition

$$\begin{array}{r}
x^3 + x^2 \qquad + 2 \\
- \;\; (x^2 - x + 1) \\
\hline
x^3 \qquad + x + 1
\end{array}$$

(b) Subtraction

$$\begin{array}{r}
x^3 + x^2 \qquad + 2 \\
\times \;\; (x^2 - x + 1) \\
\hline
x^3 + x^2 \qquad + 2 \\
-x^4 - x^3 \qquad - 2x \quad \\
x^5 + x^4 \qquad + 2x^2 \qquad\quad \\
\hline
x^5 \qquad\quad + 3x^2 - 2x + 2
\end{array}$$

(c) Multiplication

$$\begin{array}{r l}
 & x + 2 \\
\cline{2-2}
x^2 - x + 1 \,\big) & x^3 + x^2 \qquad + 2 \\
 & x^3 - x^2 + x \\
\cline{2-2}
 & 2x^2 - x + 2 \\
 & 2x^2 - 2x + 2 \\
\cline{2-2}
 & x
\end{array}$$

(d) Division

Figure 2.3 Examples of Polynomial Arithmetic

A polynomial over a field is called irreducible if and only if cannot be expressed as a product of two polynomials, both over, and both of degree lower than that of. By analogy to integers, an irreducible polynomial is also called a prime polynomial.

2.6 SYMMETRIC KEY CIPHERS

Symmetric ciphers use the same cryptographic keys for both encryption of plaintext and decryption of ciphertext. They are faster than asymmetric ciphers and allow encrypting large sets of data. However, they require sophisticated mechanisms to securely distribute the secret keys to both parties

Definition **A symmetric cipher** defined over (K, M, C), where:

- K - a set of all possible keys,
- M - a set of all possible messages,
- C - a set of all possible ciphertexts

is a pair of efficient algorithms (E, D), where:

- E: $K \times M \rightarrow C$
- D: $K \times C \rightarrow M$

such that for every m belonging to M, k belonging to K there is an equality:

- $D(k, E(k, m)) = m$ (the consistency rule)

➡ *Function E is often randomized*

➡ *Function D is always deterministic*

Types of keys are used in symmetric key cryptography

Symmetric encryption (figure 2.4) uses a single key that needs to be shared among the people who need to receive the message while asymmetrical encryption usesa pair of public key and a private key to encrypt and decrypt messages when communicating.

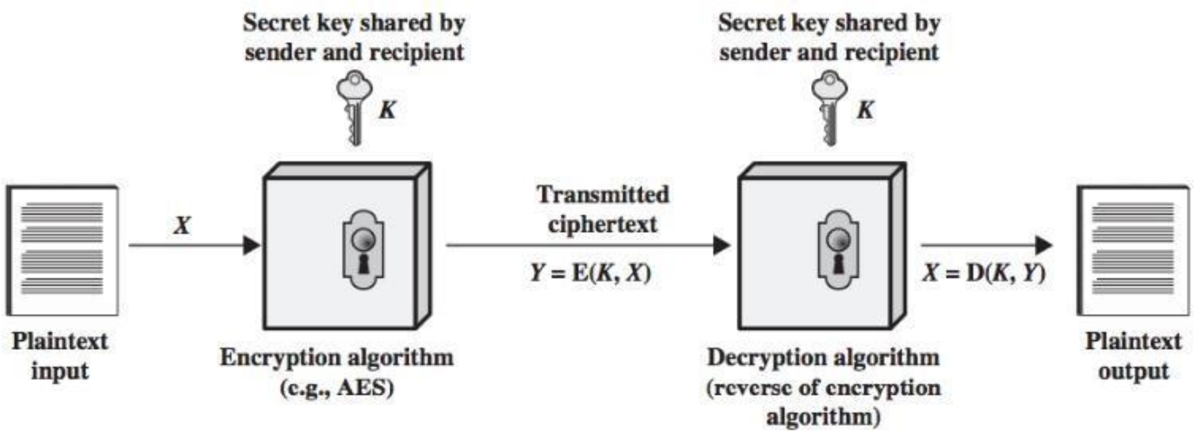


Figure 2.4 Simplified Model of Symmetric Encryption

2.7 SIMPLIFIED DATA ENCRYPTION STANDARD (S-DES)

The overall structure of the simplified DES shown in Figure 2.5. The S-DES encryption algorithm takes an 8-bit block of plaintext (example: 10111101) and a 10-bit key as input and produces an 8-bit block of ciphertext as output.

The S-DES decryption algorithm takes an 8-bit block of ciphertext and the same 10-bit key used to produce that ciphertext as input and produces the original 8-bit block of plaintext.

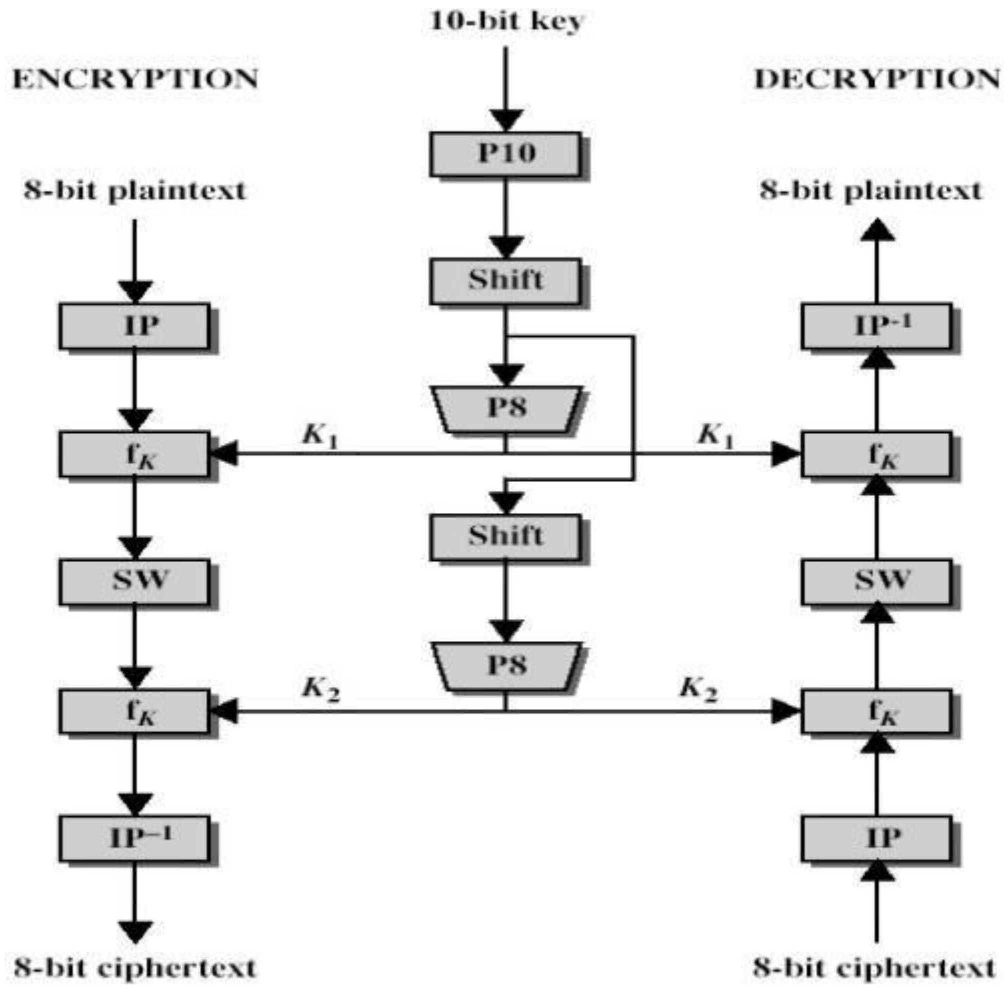


Figure 2.5 Overview of S-DES Algorithm

The encryption algorithm involves five functions:

- An initial permutation (IP)
- A complex function labeled fk, which involves both permutation and substitution operations and depends on a key input.
- A simple permutation function that switches (SW) the two halves of the data.
- The function fk again.

A permutation function that is the inverse of the initial permutation

The function fk takes as input not only the data passing through the encryption algorithm, but also an 8-bit key. Here a 10-bit key is used from which two 8-bit subkeys are generated.

The key is first subjected to a permutation (P10). Then a shift operation is performed. The output of the shift operation then passes through a permutation function that produces an 8-bit output (P8) for the first subkey (K1).

The output of the shift operation also feeds into another shift and another instance of P8 to produce the second subkey (K2).

The encryption algorithm can be expressed as a composition composition1 of functions:

IP-1 o fK2 o SW o fk1 o IP, which can also be written as

Ciphertext = IP-1 (fK2 (SW (fk1 (IP (plaintext)))))

Where

K1 = P8 (Shift (P10 (Key)))

K2 = P8 (Shift (shift (P10 (Key))))

Decryption can be shown as Plaintext = IP-1 (fK1 (SW (fk2 (IP (ciphertext)))))

2.7.2 S-DES Key Generation

S-DES depends on the use of a 10-bit key shared between sender and receiver. From this key, two 8-bit subkeys are produced for use in particular stages of the encryption and decryption algorithm.(Figure 2.6)

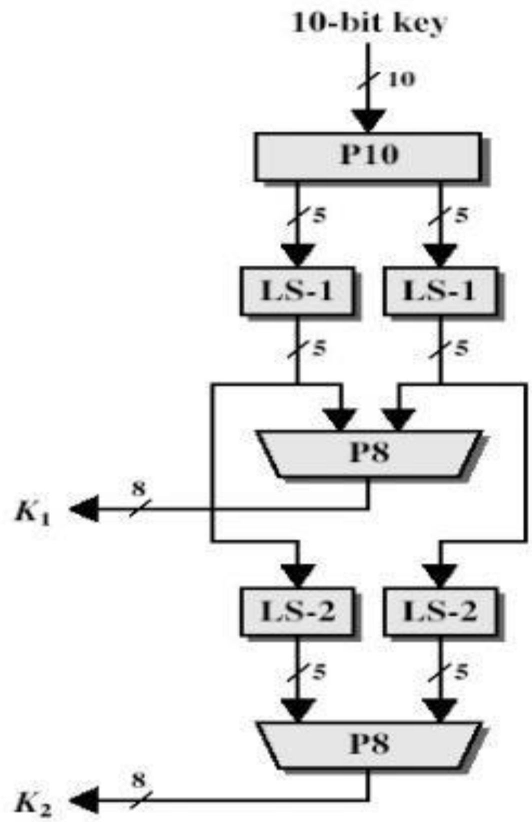


Figure 2.6 S-DES Key Generation

First, permute the key in the following fashion. Let the 10-bit key be designated as (k1, K2, k3, k4, k5, k6, k7, k8, k9, k10). Then the permutation P10 is defined as:

P10 (k1, K2, k3, k4, k5, k6, k7, k8, k9, k10) = (k3, k5, K2, k7, k4, k10 10, k1, k9, k8, k6).

P10 can be concisely defined by the display:

P10									
3	5	2	7	4	10	1	9	8	6

This table is read from left to right; each position in the table gives the identity of the input bit that produces the output bit in that position. So, the first output bit is bit 3 ofthe input; the second output bit is bit 5 of the input, and so on.

Example

The 10 bit key is (1010000010), now find the permutation from P10 for this key so it becomes (10000 01100).

Next, perform a circular left shift (LS-1), or rotation, separately on the first five bits and the second five bits. In our example, the result is (00001 11000).

Next, apply P8, which picks out and permutes 8 of the 10 bits according to the following rule:

P8							
6	3	7	4	8	5	10	9

So, The result is subkey 1 (K1). In our example, this yield (10100100).

Then go back to the pair of 5-bit strings produced by the two LS-1 functions and performs a circular left shift of 2 bit positions on each string. In our example, the value (00001 11000) becomes (00100 00011).

Finally, P8 is applied again to produce K2. In our example, the result is (01000011).

2.7.3 S-DES Encryption

Encryption involves the sequential application of five functions (Figure 2.7).

1. **Initial Permutations**

The input to the algorithm is an 8-bit block of plaintext, which we first permute using the IP function

IP							
2	6	3	1	4	8	5	7

The plaintext is 10111101

Permutated output is 01111110

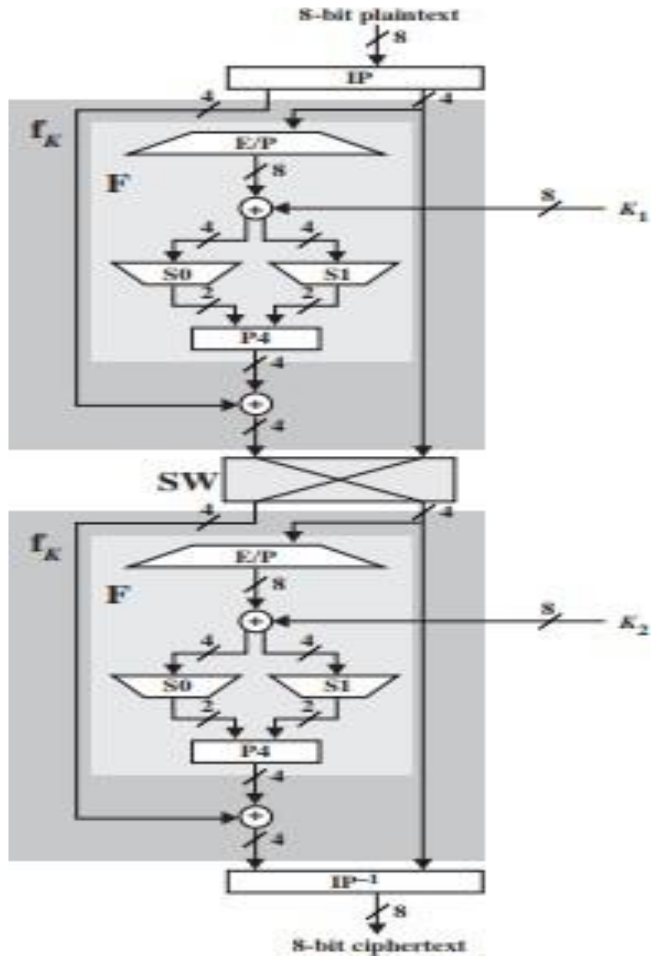


Figure 2.7 S-DES Encryption

2. The Function f_k

The most complex component of S-DES is the function fk, which consists of a combination of permutation and substitution functions. The functions can be expressed as follows. Let L and R be the leftmost 4 bits and rightmost 4 bits of the 8-bit input to f K, and let F be a mapping (not necessarily one to one) from 4-bit strings to 4-bit strings. Then we let

$$Fk\ (L, R) = (L \oplus F\ (R, SK), R)$$

Where SK is a sub key and $\oplus$is the bit-by- bit exclusive OR function

Now, describe the mapping F. The input is a 4-bit number (n1 n2 n3 n4). The first operation is an expansion/permutation operation:

E/P							
4	1	2	3	2	3	4	1

Now, find the E/P from IP

IP = 01111110, it becomes

E/P = 01111101

Now, XOR with K1

=> 01111101 $\oplus$10100100 = 11011001

The first 4 bits (first row of the preceding matrix) are fed into the S-box S0 to produce a 2- bit output, and the remaining 4 bits (second row) are fed into S1 to produce another 2-bit output.

These two boxes are defined as follows:

$$S0 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{bmatrix} \end{array} \qquad S1 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{bmatrix} \end{array}$$

The S-boxes operate as follows. The first and fourth input bits are treated as a 2- bit number that specify a row of the S-box, and the second and third input bits specify a column of the S-box. Each s box gets 4-bit input and produce 2 bits as output. It follows 00- 0, 01-1, 10-2, 11-3 scheme.

Here, take first 4 bits,

S_0 => 1101

11 - > 3

10 -> 2 => 3 =>11

So, we get 1110

Second 4 bits

S1 => 1001

11 -> 3

00 -> 0 = > 2 => 10

- Now, find P_4

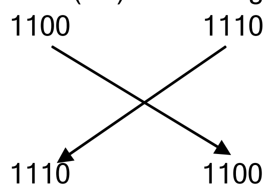
P4			
2	4	3	1

After P_4, the value is 1011

Now, XOR operation 1011⊕ 0111 => 1100

3. The Switch function

- The switch function (sw) interchanges the left and right 4 bits.



.

4. Second function f_k

- First, do E/P function and XOR with K_2, the value is 01101001⊕01000011, the answer is 00101010
- Now, find S_0 and S_1

S_0 => 00 - > 0

01 -> 1 => 0 = 00

S_1 = > 10 -> 2

01 -> 1 = > 0 => 00

Value is 0000

- Now, find P_4 and XOR operation

After P_4 => 0000 ⊕ 1110 = 1110, then concatenate last 4 bits after interchange in sw.

- Now value is 11101100

5. Find IP^{-1}

IP –1							
4	1	3	5	7	2	8	6

So, value is 01110101
The Ciphertext is 01110101

2.8.3 S-DES Decryption

- Decryption involves the sequential application of five functions.

1. **Find IP**
    - After IP, value is 11101100
2. **Function f_k**
    - After step 2, the answer is 11101100
3. **Swift**
    - The answer is 11001110
4. **Second f_k**
    - The answer is 01111110
5. **Find IP-1**
    - **101111101 -> Plaintext**

2.8 DATA ENCRYPTION STANDARD

The most widely used encryption scheme is based on the Data Encryption Standard (DES) adopted in 1977. The algorithm itself is referred to as the Data Encryption Algorithm (DEA).

For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output.

2.8.1 DES Encryption

The overall scheme for DES encryption is illustrated in the Figure 2.8. There are two inputs to the encryption function: the **plaintext** to be encrypted and the **key**. The plaintext must be 64 bits in length and the key is 56 bits in length.

2.8.2 General Depiction of DES Encryption Algorithm

Phase 1

Looking at the left-hand side of the figure 2.8, we can see that the processing of the plaintext proceeds in three phases.

First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the *permuted input*.

Phase 2:

This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions.

The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the preoutput.

Phase 3:

Finally, the preoutput is passed through a permutation (IP^{-1}) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.
The right-hand portion of Figure shows the way in which the 56-bit key is used.

Operation on key:

Initially, the key is passed through a permutation function. Then, for each of the 16 rounds, a *subkey* (*Ki*) is produced by the combination of a left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is producedbecause of the repeated shifts of the key bits.

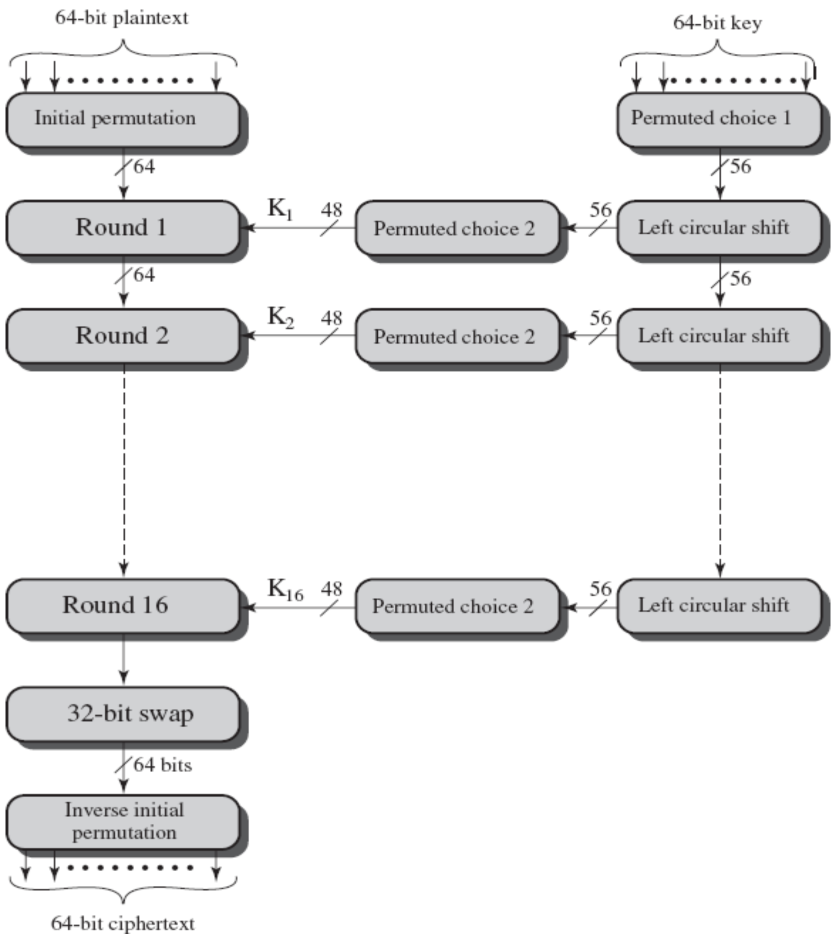


Figure 2.8 DES Encryption Algorithm

Initial Permutation

The input to a table consists of 64 bits numbered from 1 to 64. The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64. Each entry in the permutation table indicates the position of a numbered input bit in the output, which alsoconsists of 64 bits.

Permutation Tables for DES

(a) Initial Permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Inverse Initial Permutation (IP^{-1})

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Expansion Permutation (E)

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Permutation Function (P)

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Consider the following 64-bit input *M*:

$M1$	$M2$	$M3$	$M4$	$M5$	$M6$	$M7$	$M8$
$M9$	$M10$	$M11$	$M12$	$M13$	$M14$	$M15$	$M16$
$M17$	$M18$	$M19$	$M20$	$M21$	$M22$	$M23$	$M24$
$M25$	$M26$	$M27$	$M28$	$M29$	$M30$	$M31$	$M32$
$M33$	$M34$	$M35$	$M36$	$M37$	$M38$	$M39$	$M40$
$M41$	$M42$	$M43$	$M44$	$M45$	$M46$	$M47$	$M48$
$M49$	$M50$	$M51$	$M52$	$M53$	$M54$	$M55$	$M56$
$M57$	$M58$	$M59$	$M60$	$M61$	$M62$	$M63$	$M64$

where *Mi* is a binary digit. Then the permutation $X = \text{IP}(M)$ is as follows:

$M58$	$M50$	$M42$	$M34$	$M26$	$M18$	$M10$	$M2$
$M60$	$M52$	$M44$	$M36$	$M28$	$M20$	$M12$	$M4$
$M62$	$M54$	$M46$	$M38$	$M30$	$M22$	$M14$	$M6$
$M64$	$M56$	$M48$	$M40$	$M32$	$M24$	$M16$	$M8$
$M57$	$M49$	$M41$	$M33$	$M25$	$M17$	$M9$	$M1$
$M59$	$M51$	$M43$	$M35$	$M27$	$M19$	$M11$	$M3$
$M61$	$M53$	$M45$	$M37$	$M29$	$M21$	$M13$	$M5$
$M63$	$M55$	$M47$	$M39$	$M31$	$M23$	$M15$	$M7$

Inverse permutation $Y = \text{IP}^{-1}(X) = \text{IP}^{-1}(\text{IP}(M))$,Therefore we can see that the original ordering of the bits is restored.

2.8.3 Details of Single Round

The below figure 2.9 shows the internal structure of a single round. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right). The overall processing at each round can be summarized in the following formulas:

$L_i = R_{i-1}$

$R_i = L_{i-1} \text{ x } F(R_{i-1}, K_i)$

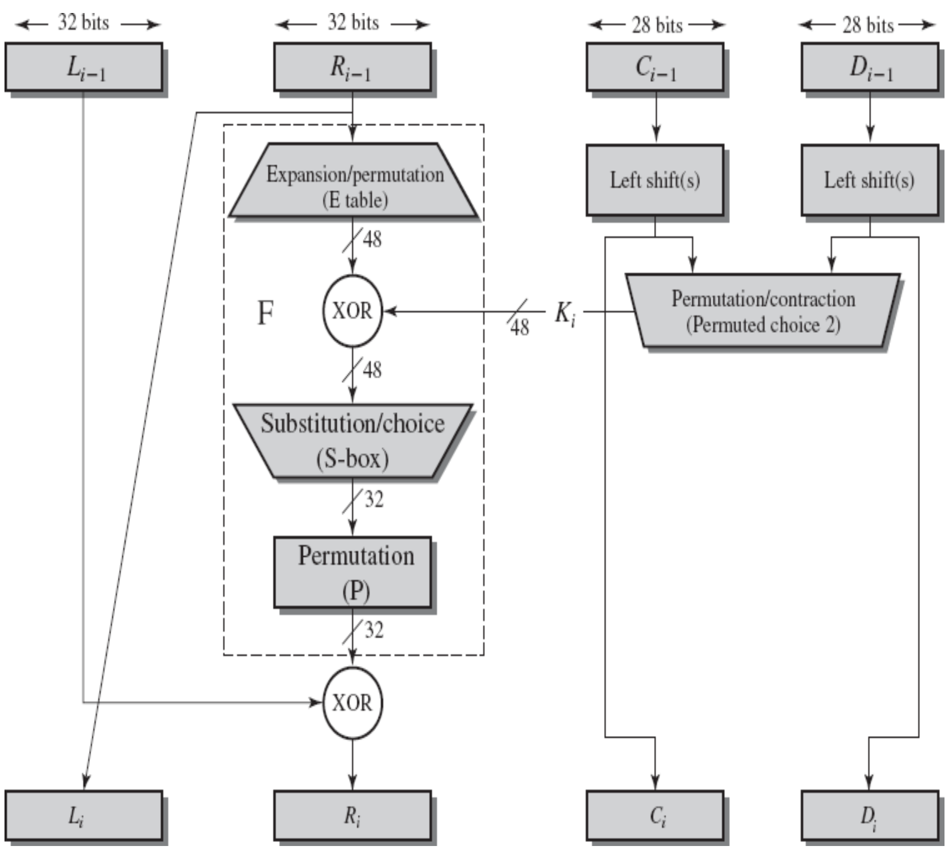


Figure 2.9 Single Round of DES Algorithm

The round key *Ki* is 48 bits. The *R* input is 32 bits. This *R* input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the *R* bits. The resulting 48 bits are XORed with *Ki*. This 48-bit result passes through a substitution function that produces a 32-bit output, which is then permuted.

Definition of S-Boxes

The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. The first and last bits of the input to box S*i* form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for S*i*. The middle four bits select one of the sixteen columns as shown in figure 2.10.

The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output.

For example, in S1 for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.

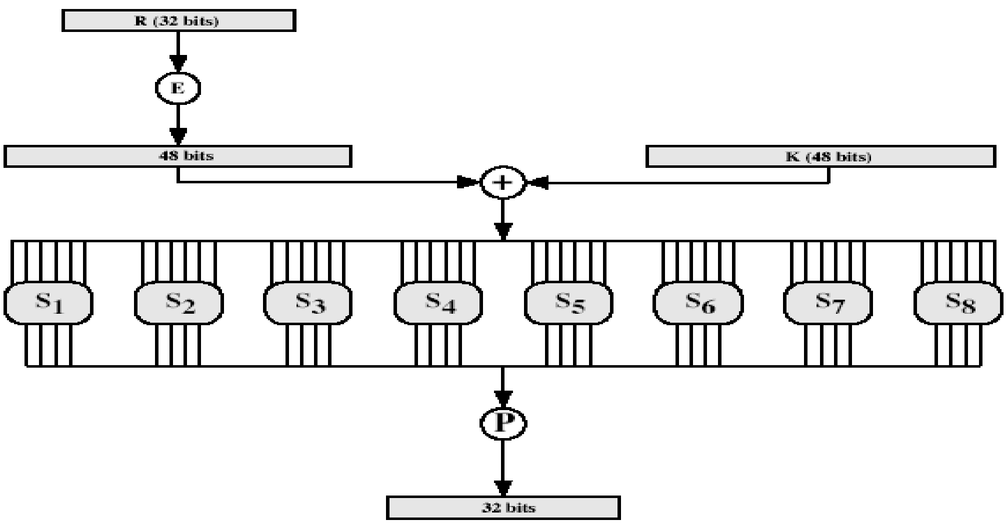


Fig 2.10 Calculation of F(R, K)

2.8.4 Key Generation

The 64-bit key is used as input to the algorithm. The bits of the key are numbered from 1 through 64; every eighth bit is ignored. The key is first subjected to a permutation governed by a table labeled Permuted Choice One. The resulting 56-bit key is then treated as two 28-bit quantities, labeled *C*0 and *D*0.

At each round, *Ci*-1 and *Di*-1 are separately subjected to a circular left shift, or rotation, of 1 or 2 bits. These shifted values serve as input to the next round. They also serve as input to Permuted Choice 2, which produces a 48-bit output that serves as input to the function F(*Ri*-1, *Ki*).

DES Key Schedule Calculation

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (PC-1)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

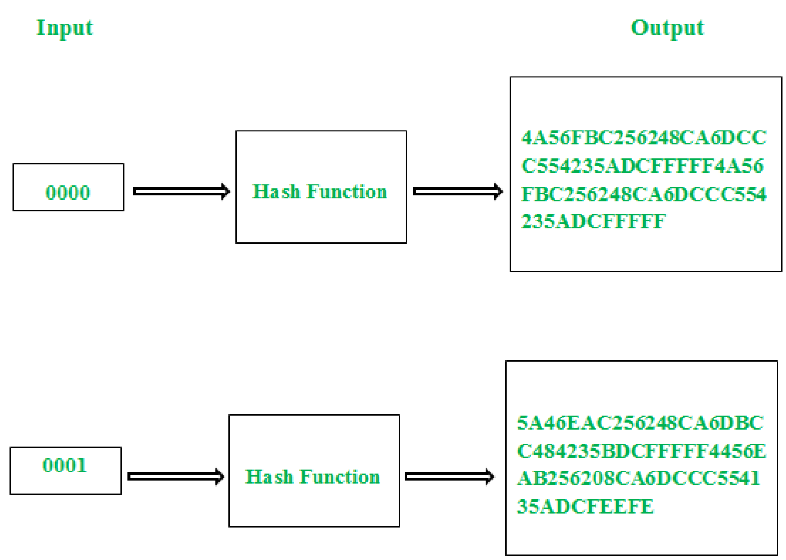
Roundnumber:1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Bits rotated : 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1

2.8.5 DES Decryption:

As with any Feistel cipher, decryption uses the same algorithm as encryption,except that the application of the subkeys is reversed. Additionally, the initial andfinal permutations are reversed.

2.8.6 The Avalanche Effect:

A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext.



2.9 THE STRENGTH OF DES

The strength of DES depends on two factors: **key size** and the **nature of the algorithm**.

1. **The Use of 56-Bit Keys**
   With a key length of 56 bits, there are 2^{56} possible keys, which is approximately 7.2×10^{16}. Thus, a brute-force attack appears impractical.

2. **The Nature of the DES Algorithm**
   In DES algorithm, eight substitution boxes called S-boxes that are used in each iteration. Because the design criteria for these boxes, and indeed for the entire algorithm, were not made public, there is a suspicion that the boxes were constructed in such a way that cryptanalysis is possible for an opponent who knows the weaknesses in the S-boxes. Despite this, no one hasso far succeeded in discovering the supposed fatal weaknesses in the S-boxes.

3. **Timing Attacks**
   A timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts. A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs.

2.9.1 Attacks on DES:

Two approaches are:

1. Differential crypt analysis
2. Linear crypt analysis