**PANDIT DEENDAYAL ENERGY UNIVERSITY**

****

**Submitted By:**

**Name:** Dhyey Shekhaliya

**Roll Number:** 22BCP301

**Division:** 4

**Group:** 8

**Submitted To: Professor Sujit Kumar Das**

**Experiment-1: Implementation of S-DES**

**Theory**

**S-DES** is a reduced version of the DES (Data Encryption Standard) algorithm, designed for educational purposes to demonstrate the core principles of DES. It uses a smaller block size, fewer rounds, and simpler key schedules. Here's an explanation of its key components:

**Key Concepts of S-DES**

1. **Plaintext and Ciphertext**:
   * Block size: 8 bits.
   * Input plaintext is 8 bits long.
2. **Key**:
   * 10-bit key input.
   * From this, two 8-bit subkeys (K1 and K2) are generated through permutation and shifts.
3. **Encryption**:
   * 8-bit plaintext goes through an Initial Permutation (IP), two rounds of a Feistel function, and finally an inverse permutation (IP⁻¹) to produce ciphertext.
4. **Decryption**:
   * Reverses the encryption process using the same subkeys (but in reverse order).
5. **Feistel Function (Fk)**:
   * Involves splitting the 8-bit input into two 4-bit halves.
   * The right half is expanded/permuted and XORed with a subkey.
   * The result is passed through two S-boxes (S0 and S1).
   * The S-box outputs are permuted and XORed with the left half.
6. **S-Boxes**:
   * S0 and S1 are small substitution boxes used for nonlinear substitution.

**Algorithm Steps**

**1. Key Generation**

* **Input**: 10-bit key.
* **Steps**:
  1. Perform a Permutation (P10).
  2. Split into two 5-bit halves.
  3. Perform a left circular shift (LS-1) on each half.
  4. Apply Permutation (P8) to generate **K1**.
  5. Perform another left circular shift (LS-2) on the halves.
  6. Apply Permutation (P8) to generate **K2**.

**2. Encryption**

* **Input**: 8-bit plaintext and subkeys (K1, K2).
* **Steps**:
  1. Perform Initial Permutation (IP).
  2. Split the 8-bit result into two 4-bit halves.
  3. Apply Feistel function Fk with **K1**.
  4. Swap the left and right halves.
  5. Apply Feistel function Fk with **K2**.
  6. Perform Inverse Permutation (IP⁻¹).

**3. Decryption**

* **Input**: 8-bit ciphertext and subkeys (K1, K2).
* **Steps**:
  1. Perform Initial Permutation (IP).
  2. Split the 8-bit result into two 4-bit halves.
  3. Apply Feistel function Fk with **K2** (reverse key order).
  4. Swap the left and right halves.
  5. Apply Feistel function Fk with **K1**.

**Code:**

**Output:**

**Conclusion:**