## Chapter 3 The Data Encryption Standard

### 3.1 Introduction

On May 15, 1973, the National Bureau of Standards published a solicitation for cryptosystems in the Federal Register. This lead ultimately to the development of the **Data Encryption Standard**, or **DES**, which has become the most widely used cryptosystem in the world. **DES** was developed at IBM, as a modification of an earlier system known as **LUCIFER. DES** was first published in the Federal Register of March 17, 1975. After a considerable amount of public discussion, **DES** was adopted as a standard for “unclassified” applications on January 15, 1977. **DES** has been reviewed by the National Bureau of Standards (approximately) every five years since its adoption. Its most recent renewal was in January 1994, when it was renewed until 1998. It is anticipated that it will not remain a standard past 1998.

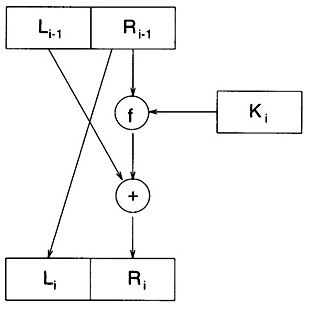
### 3.2 Description of DES

A complete description of **DES** is given in the Federal Information Processing Standards Publication 46, dated January 15, 1977. **DES** encrypts a plaintext bitstring *x* of length 64 using a key *K* which is a bitstring of length 56, obtaining a ciphertext bitstring which is again a bitstring of length 64. We first give a “high-level” description of the system.

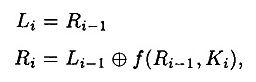
The algorithm proceeds in three stages:

**1.**  Given a plaintext *x*, a bitstring *x*0 is constructed by permuting the bits of *x* according to a (fixed) *initial permutation* IP. We write *x*0 = IP (*x*) = *L*0*R*0, where *L*0 comprises the first 32 bits of *x*0 and *R*0 the last 32 bits.

**2.**  16 iterations of a certain function are then computed. We compute *LiRi*,

  
[**Figure 3.1**](javascript:displayWindow('images/03-01.jpg',300,297))  One round of DES encryption

1 ≤ *i* ≤ 16, according to the following rule:

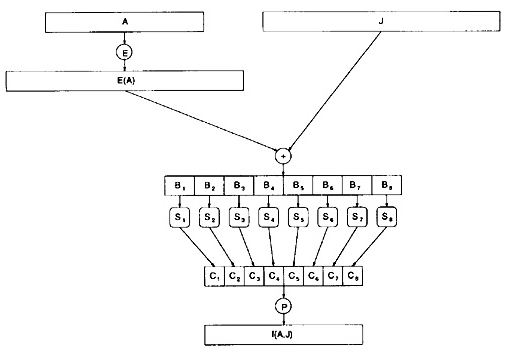


where ⊕ denotes the exclusive-or of two bitstrings. *f* is a function that we will describe later, and *K*1, *K*2, . . . , *K*16 are each bitstrings of length 48 computed as a function of the key *K*. (Actually, each *Ki* is a permuted selection of bits from *K*.) *K*1, *K*2, . . . , *K*16 comprises the *key schedule*. One round of encryption is depicted in Figure 3.1

**3.**  Apply the inverse permutation IP-1 to the bitstring *R*16*L*16, obtaining the ciphertext *y*. That is, *y* = IP-1(*R*16*L*16). Note the inverted order of *L*16 and *R*16.

The function *f* takes as input a first argument *A*, which is a bitstring of length 32, and a second argument *J* that is a bitstring of length 48, and produces as output a bitstring of length 32. The following steps are executed.

**1.**  The first argument *A* is “expanded” to a bitstring of length 48 according to a fixed *expansion function* E. E(*A*) consists of the 32 bits from *A*, permuted in a certain way, with 16 of the bits appearing twice.

  
[**Figure 3.2**](javascript:displayWindow('images/03-02.jpg',500,338))  The DES *f* function

**2.**  Compute E(*A*) ⊕ *J* and write the result as the concatenation of eight 6-bit strings *B* = *B*1*B*2*B*3*B*4*B*5*B*6*B*7*B*8.

**3.**  The next step uses eight *S-boxes* *S*1, . . . , *S*8. Each *Si* is a fixed 4 × 16 array whose entries come from the integers 0 - 15. Given a bitstring of length six, say *Bj* = *b*1*b*2*b*3*b*4*b*5*b*6, we compute *Sj*(*Bj*) as follows. The two bits *b*1*b*6 determine the binary representation of a row *r* of *Sj* (0 ≤ *r* ≤ 3), and the four bits *b*2*b*3*b*4*b*5 determine the binary representation of a column *c* of *Sj* (0 ≤ *c* ≤ 15). Then *Sj*(*Bj*) is defined to be the entry *Sj*(*r, c*), written in binary as a bitstring of length four. (Hence, each *Sj* can be thought of as a function that accepts as input a bitstring of length two and one of length four, and produces as output a bitstring of length four.) In this fashion, we compute *Cj* = *Sj*(*Bj*), 1 ≤ *j* ≤ 8.

**4.**  The bitstring *C* = *C*1*C*2*C*3*C*4*C*5*C*6*C*7*C*8 of length 32 is permuted according to a fixed permutation P. The resulting bitstring P(*C*) is defined to be *f*(*A, J*).

The *f* function is depicted in Figure 3.2. Basically, it consists of a substitution (using an S-box) followed by the (fixed) permutation P. The 16 iterations of *f* comprise a product cryptosystem, as described in Section 2.5.

In the remainder of this section, we present the specific functions used in **DES**.

The initial permutation IP is as follows:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |

This means that the 58th bit of *x* is the first bit of IP(*x*); the 50th bit of *x* is the second bit of IP(*x*), etc.

The inverse permutation IP-1 is:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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 |

The expansion function E is specified by the following table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **E bit-selection table** | | | | | |
| 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 |

The eight S-boxes and the permutation P are now presented:

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

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

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

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

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

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

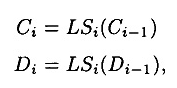
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***S*8** | | | | | | | | | | | | | | | |
| 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12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

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |

Finally, we need to describe the computation of the key schedule from the key *K*. Actually, *K* is a bitstring of length 64, of which 56 bits comprise the key and 8 bits are parity-check bits (for error-detection). The bits in positions 8, 16, . . . , 64 are defined so that each byte contains an odd number of 1’s. Hence, a single error can be detected within each group of 8 bits. The parity-check bits are ignored in the computation of the key schedule.

**1.**  Given a 64-bit key *K*, discard the parity-check bits and permute the remaining bits of *K* according to a (fixed) permutation PC-1. We will write PC-1 (*K*) = *C*0*D*0, where *C*0 comprises the first 28 bits of PC-1(*K*) and *D*0 the last 28 bits.

**2.**  For *i* ranging from 1 to 16, compute

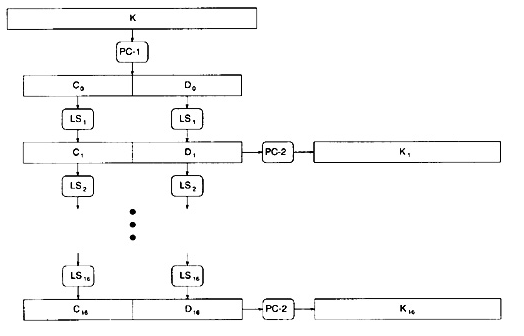


and *Ki* = PC-2(*CiDi*). *LSi* represents a cyclic shift (to the left) of either one or two positions, depending on the value of *i*: shift one position if *i* = 1, 2, 9 or 16, and shift two positions otherwise. PC-2 is another fixed permutation.

The key schedule computation is depicted in Figure 3.3.

The permutations PC-1 and PC-2 used in the key schedule computation are as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

  
[**Figure 3.3**](javascript:displayWindow('images/03-03.jpg',500,316))  Computation of DES key schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 |

We now display the resulting key schedule. As mentioned above, each round uses a 48-bit key comprised of 48 of the bits in *K*. The entries in the tables below refer to the bits in *K* that are used in the various rounds.

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

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

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

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

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

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

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

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Round 8** | | | | | | | | | | | |
| 36 | 41 | 60 | 50 | 10 | 43 | 59 | 18 | 57 | 35 | 9 | 3 |
| 58 | 25 | 52 | 51 | 34 | 19 | 49 | 27 | 26 | 17 | 44 | 2 |
| 12 | 54 | 61 | 13 | 31 | 30 | 6 | 20 | 62 | 47 | 45 | 23 |
| 55 | 15 | 28 | 22 | 37 | 46 | 39 | 4 | 7 | 21 | 14 | 53 |

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

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

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

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

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

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

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

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Round 16** | | | | | | | | | | | |
| 18 | 59 | 42 | 3 | 57 | 25 | 41 | 36 | 10 | 17 | 27 | 50 |
| 11 | 43 | 34 | 33 | 52 | 1 | 2 | 9 | 44 | 35 | 26 | 49 |
| 30 | 5 | 47 | 62 | 45 | 12 | 55 | 38 | 13 | 61 | 31 | 37 |
| 6 | 29 | 46 | 4 | 23 | 28 | 53 | 22 | 21 | 7 | 63 | 39 |

Decryption is done using the same algorithm as encryption, starting with *y* as the input, but using the key schedule *K*16, . . . , *K*1 in reverse order. The output will be the plaintext *x*.

**3.2.1 An Example of DES Encryption**

Here is an example of encryption using the **DES**. Suppose we encrypt the (hexadecimal) plaintext

0123456789ABCDEF

using the (hexadecimal) key

133457799BBCDFF1.

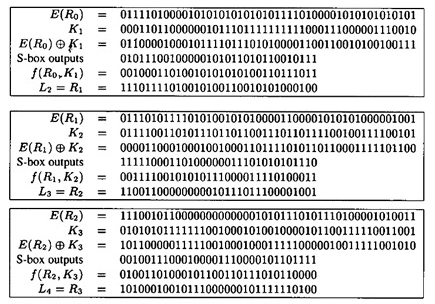
The key, in binary, without parity-check bits, is

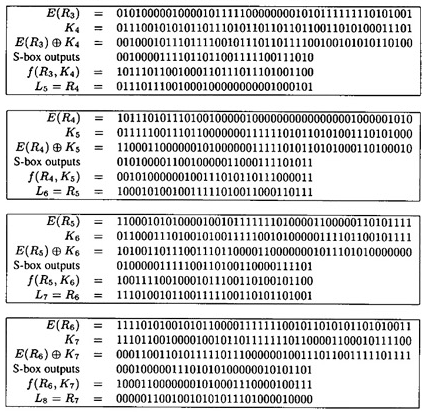
00010010011010010101101111001001101101111011011111111000.

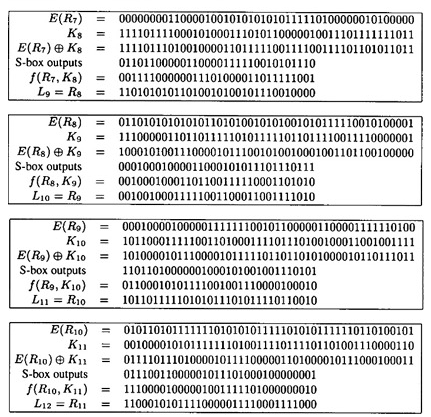
Applying IP, we obtain *L*0 and *R*0 (in binary):

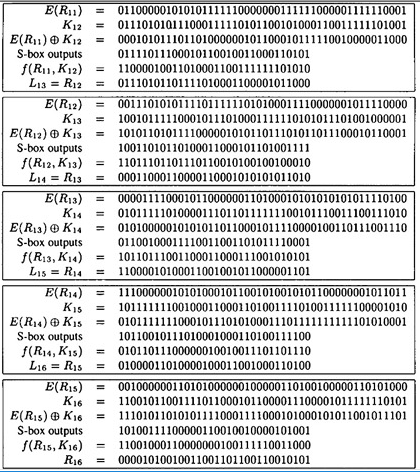


The 16 rounds of encryption are then performed, as indicated.









Finally, applying IP-1 to *R*16*L*16, we obtain the ciphertext, which (in hexadecimal form) is:

85E813540F0AB405.