# Block Ciphers and Data Encryption Standard

CSS 322 - Security and Cryptography

# Contents

- Block Cipher Principles
- Feistel Structure for Block Ciphers
- DES
  - Simplified DES
  - Real DES
- DES Design Issues

# Block and Stream Ciphers

- Block cipher
  - Encryption algorithm is applied on a fixed size block of information
    - If block too short (e.g. 8 bits) then easy to break if obtain some plaintext, ciphertext pairs
    - If block too long, too complex and inefficient performance
    - Typical size is 64 bits
  - But messages are not usually 64 bits!
    - Block cipher modes (another lecture) allow block encryption algorithm to be applied to many 64 bit blocks
- Stream cipher
  - Encrypt a bit at a time
  - Useful for operating in real-time applications

# Ideal Block Ciphers

- Operates on plaintext block of n bits
- Produces ciphertext block of n bits
- $2^n$ different plaintext blocks
  - In order to do decryption, each plaintext must produce unique ciphertext, i.e. must be reversible

Reversible Mapping

| Plaintext | Ciphertext |
|---|---|
| 00 | 11 |
| 01 | 10 |
| 10 | 00 |
| 11 | 01 |

Irreversible Mapping

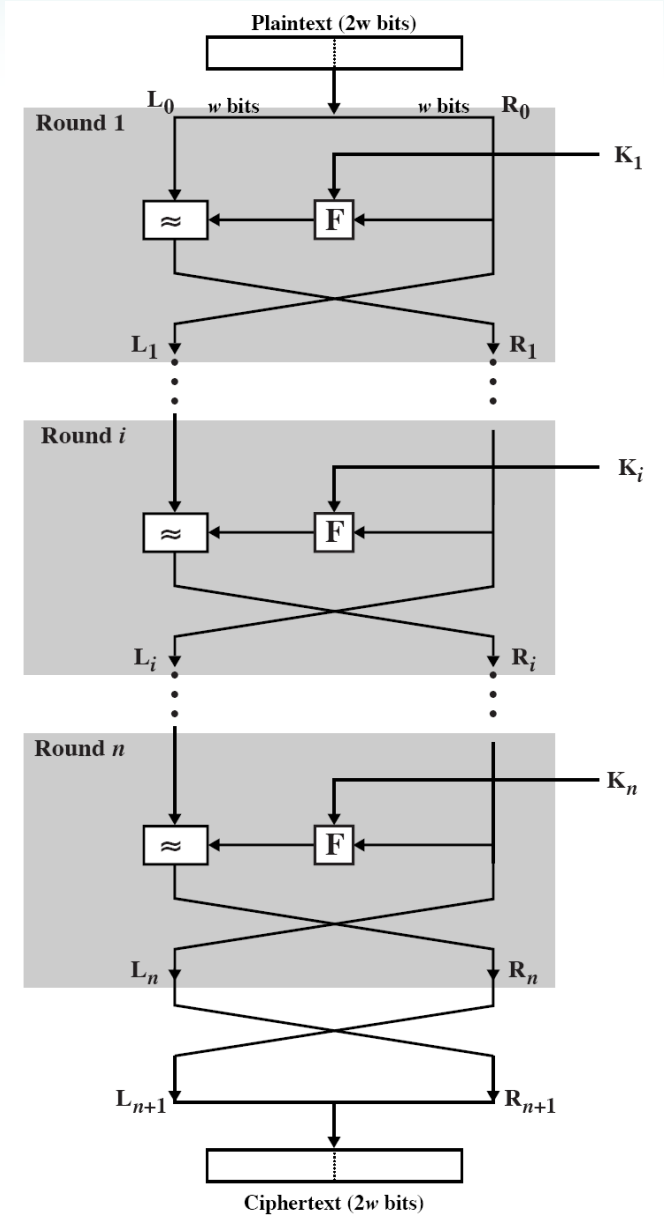
| Plaintext | Ciphertext |
|---|---|
| 00 | 11 |
| 01 | 10 |
| 10 | 01 |
| 11 | 01 |

- A n-bit input maps to $2^n$ possible input states, the substitution is used to produce $2^n$ output states, and then mapped to n-bit output
- But:
  - If n is small, easy to use statistical analysis of letters etc.
  - If n is large, then implementation and performance is too complex

# Feistel Structure for Block Ciphers

- In 1975 Feistel proposed a generic cipher that approximates ideal block cipher, but easier to implement
  - Execute two or more simple ciphers in sequence, so that result is cryptographically stronger than the simple ciphers
  - The cipher alternates substitutions and permutations
- Based on concepts of:
  - **Diffusion**: statistical nature of plaintext is reduced in ciphertext, e.g. a plaintext letter affects the value of many ciphertext letters
  - **Confusion**: Make relationship between ciphertext and key as complex as possible, e.g. a complex substitution algorithm

# Feistel Structure




- Plaintext split into halves
- Subkeys generated from Key
- Apply a round function (F) to right half
  - Subkey is parameter
- Apply substitution on left half by XOR with output of F
- Apply permutation by interchanging the halves
- Repeat rounds
- Final output of left and right halves is the ciphertext

# Using the Feistel Structure

- The exact implementation of algorithm based on Feistel structure depends on:
  - Block size: typical is 64 bits (AES uses 128 bits)
    - Larger value leads to more diffusion
  - Key size: 128 bits is common
    - Larger value leads to more confusion and resistance against brute-force attacks
  - Number of rounds: one round is insecure; 16 rounds is common
  - Subkey generation algorithm: should be complex!
  - Round function: should be complex!
- The above have tradeoffs between security and performance
  - More complex and larger leads to lower performance

# Data Encryption Standard

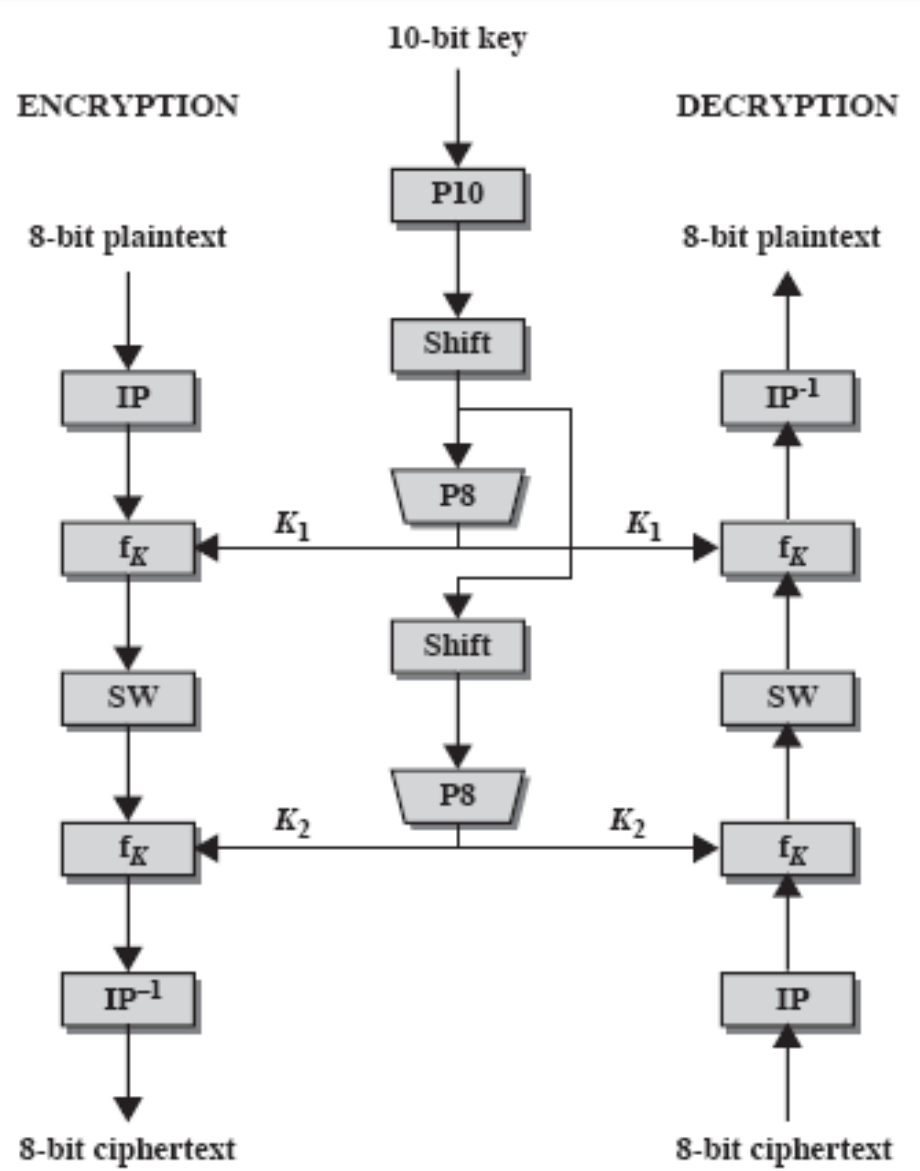
- Symmetric block cipher
  - 56-bit key
  - 64-bit input block
  - 64-bit output block
- One of most used encryption systems in world
- Principles used in DES are used in other systems as well (e.g. 3DES, IDEA)
- We will go through details of operation
  - But first will use Simplified DES as an example of main concepts
  - Note Simplified DES is not used in real world, only developed to help explain DES to students

# History of DES

- Published as standard by National Institute of Standards and Technology (NIST) in 1977 (NIST was NBS)
- Designed by IBM (based on Lucifer) with input from NSA
- Why 56-bit key?
  - Actually 64 bits but 8 bits used for parity (error check)
  - Critics suggest 56-bits was just weak enough for NSA to break
- Efficient to implement in hardware, was slow in software
  - Allow more control on implementation and distribution
  - Now feasible to implement in software
- Lack of insight into design of S-Boxes; critics believed NSA may know vulnerabilities so can break DES
- In 1999, NIST recommended Triple DES be used for new systems (DES only for legacy systems)

# Simplified DES (S-DES)

- Plaintext: 8-bit block
- Key: 10-bits
- Ciphertext: 8-bits
- Encryption steps:
  - Initial permutation (IP)
  - Function $f_k$
  - Switch (SW) halves of data
  - Final permutation ($IP^{-1}$)
- Creating keys:
  - Permutation (P10)
  - Shift bits
  - Permutation (P8)



# S-DES Key Generation

- 10-bit key is shared between sender and receiver
- Two 8-bit keys are generated from original 10-bit key and used in encryption and decryption: K1 and K2
  - P10: rearrange bits of 10-bit key

```
Input order : 1 2 3 4 5 6 7 8 9 10
Output order: 3 5 2 7 4 10 1 9 8 6
```

  - LS1: left shift bits 1 position separately on left and right five bits

```
Input order : 1 2 3 4 5 6 7 8 9 10
Output order: 2 3 4 5 1 7 8 9 10 6
```

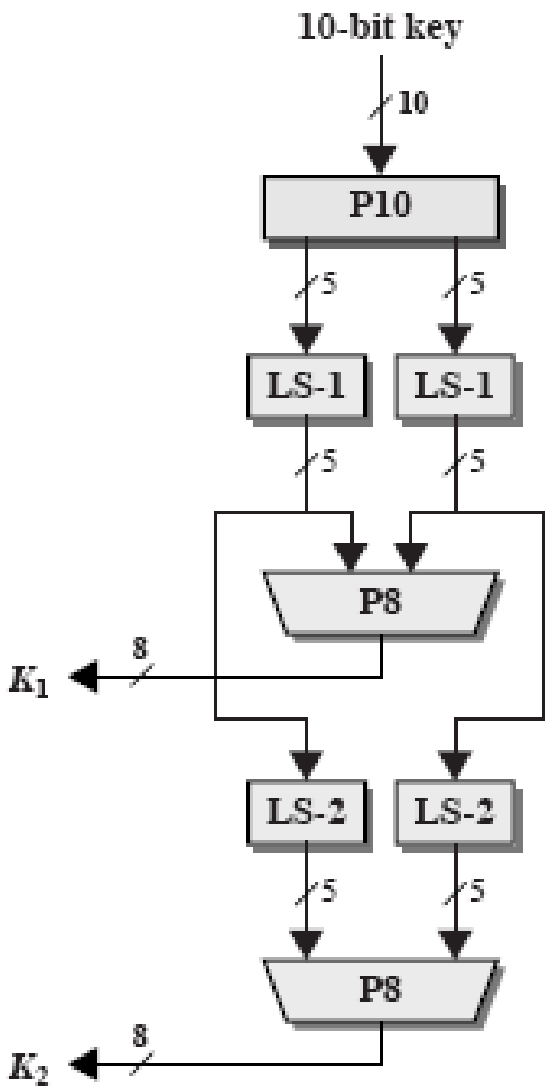
  - P8: pick 8 of 10 bits and re-arrange

```
Input order : 1 2 3 4 5 6 7 8 9 10
Output order: 6 3 7 4 8 5 10 9
```

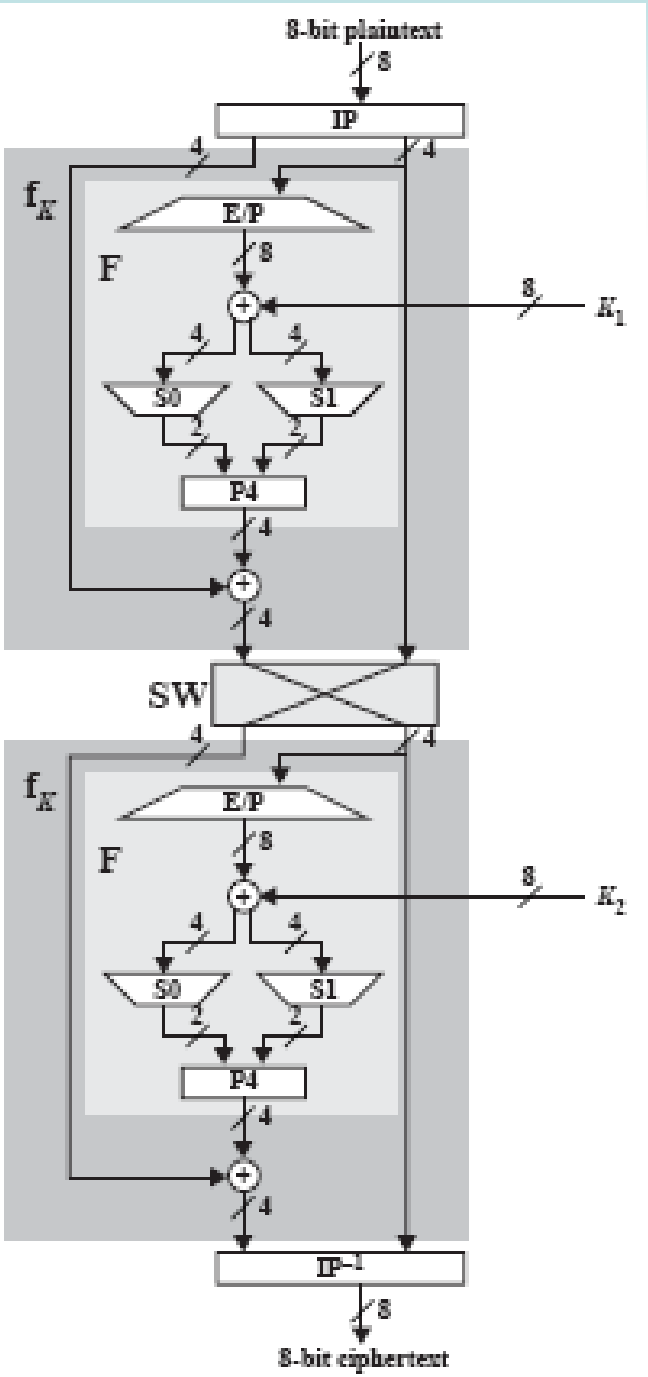
  - LS2: left shift bits 2 positions separately on left and right 5 bits

```
Input order : 1 2 3 4 5 6 7 8 9 10
Output order: 3 4 5 1 2 8 9 10 6 7
```

# S-DES Key Generation




- Example key: 1010000010
- After P10: 1000001100
- After LS1: 00001 11000
- K1 (after P8): 10100100
- After LS2: 00100 00011
- K2 (after P8): 01000011




# S-DES Encryption

- IP: rearrange 8-bits
  `Output order: 2 6 3 1 4 8 5 7`
- $IP^{-1}$: inverse of IP at end
  `Output order: 4 1 3 5 7 2 8 6`
- SW: Switch left bits with right bits
  `Output order: 5 6 7 8 1 2 3 4`
- Function $f_k$
  - Split input 8 bits into left 4 bits (L) and right 4 bits (R)
  - E/P: Expand and re-arrange R
    `Output order: 4 1 2 3 2 3 4 1`
  - XOR output of E/P with K1
  - Feed left 4 bits into S-Box S0 and right 4 bits into S-Box S1

# S-Boxes in S-DES

- Each S-Box has a 4 bit input: $b_1$, $b_2$, $b_3$, $b_4$
  - $(b_1,b_4)$ specify the row, $r$, of the S-Box
  - $(b_2,b_3)$ specify the column, $c$, of the S-Box
  - Output is the entry at $(r, c)$

$$S0 = \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} 01 & 00 & 11 & 10 \\ 11 & 10 & 01 & 00 \\ 00 & 10 & 01 & 11 \\ 11 & 01 & 11 & 10 \end{bmatrix} \end{array}$$

$$S1 = \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} 00 & 01 & 10 & 11 \\ 10 & 00 & 01 & 11 \\ 11 & 00 & 01 & 00 \\ 10 & 01 & 00 & 11 \end{bmatrix} \end{array}$$

- The 4 bits produced by S0 and S1 are re-arranged (P4):

```
Output order: 2 4 3 1
```

# S-DES Encryption and Decryption

- The output of $f_k$ is the XOR of L and output of P4
- The output of $f_k$ and R are input to the switch SW
- Then repeat the entire process but with key $K_2$
- Finally the output is input to $IP^{-1}$, producing the 8-bit ciphertext
- Decryption:
  - Follow same steps as encryption except use $K_2$ first, and then $K_1$

# Summary of S-DES

- Educational encryption algorithm
- S-DES expressed as functions:

$$ciphertext = IP^{-1}\left(f_{K_2}\left(SW\left(f_{K_1}\left(IP\left(plaintext\right)\right)\right)\right)\right)$$

$$plaintext = IP^{-1}\left(f_{K_1}\left(SW\left(f_{K_2}\left(IP\left(ciphertext\right)\right)\right)\right)\right)$$

- Security of S-DES
  - 10-bit key, 1024 keys – easy to try every key and analyse plaintext
  - If know plaintext and corresponding ciphertext, can we determine the key?
    - Very hard!

# Comparing DES and S-DES

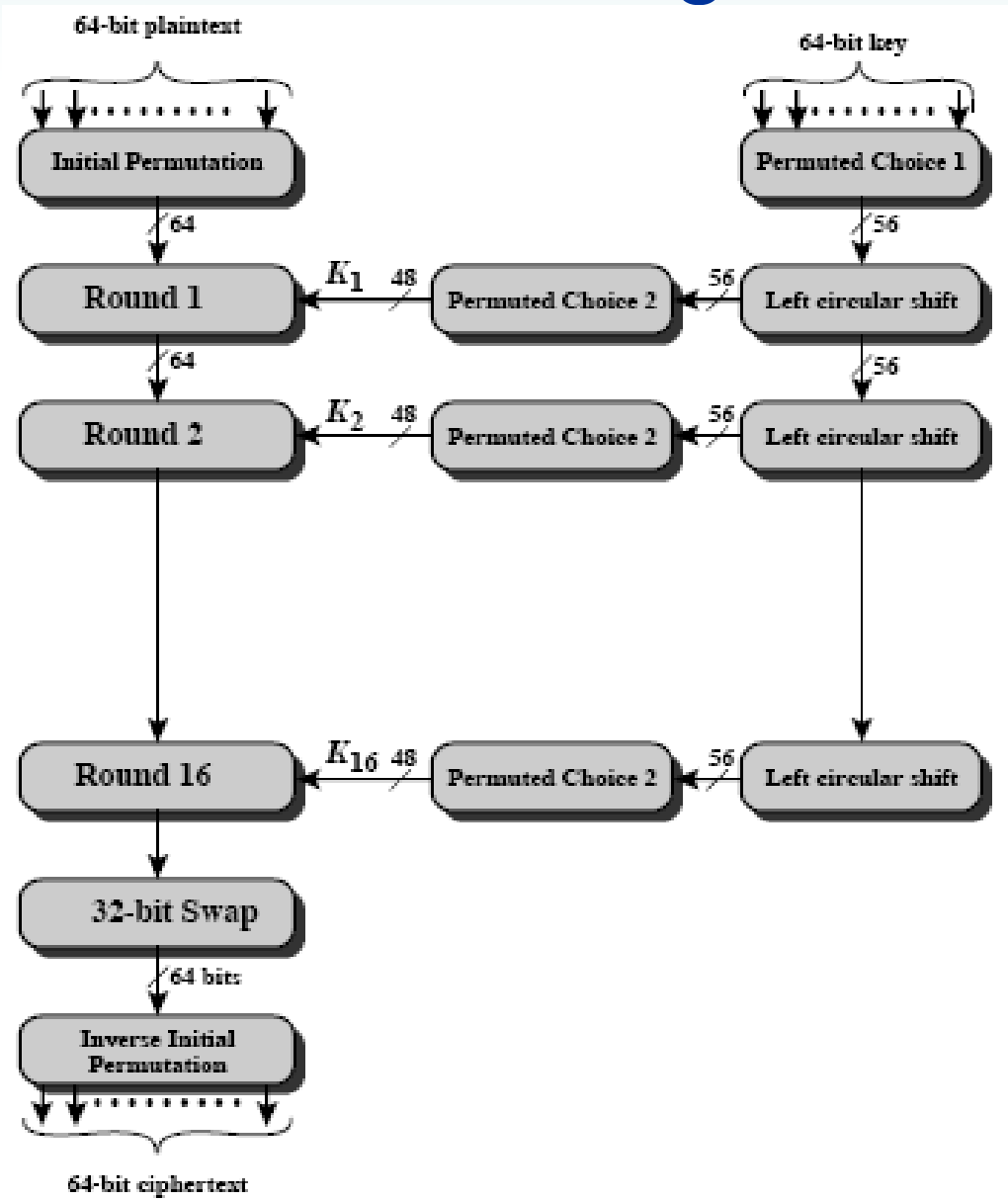
- DES
- Operates on 64-bit blocks
- 56-bit key used
  - 16 x 48-bit keys are calculated from this 56-bit key
- Initial Permutation, IP of 64 bits
- F acts on 32 bits
- 8 S-Boxes
- 16 rounds

- S-DES
- Operates on 8-bits
- 10-bit key
  - 2 x 8-bit keys are calculated from 10-bit key
- IP of 8 bits
- F acts on 4 bits
- 2 S-Boxes
- 2 rounds

S-DES $$ciphertext = IP^{-1}\left(f_{K_2}\left(SW\left(f_{K_1}\left(IP\left(plaintext\right)\right)\right)\right)\right)$$

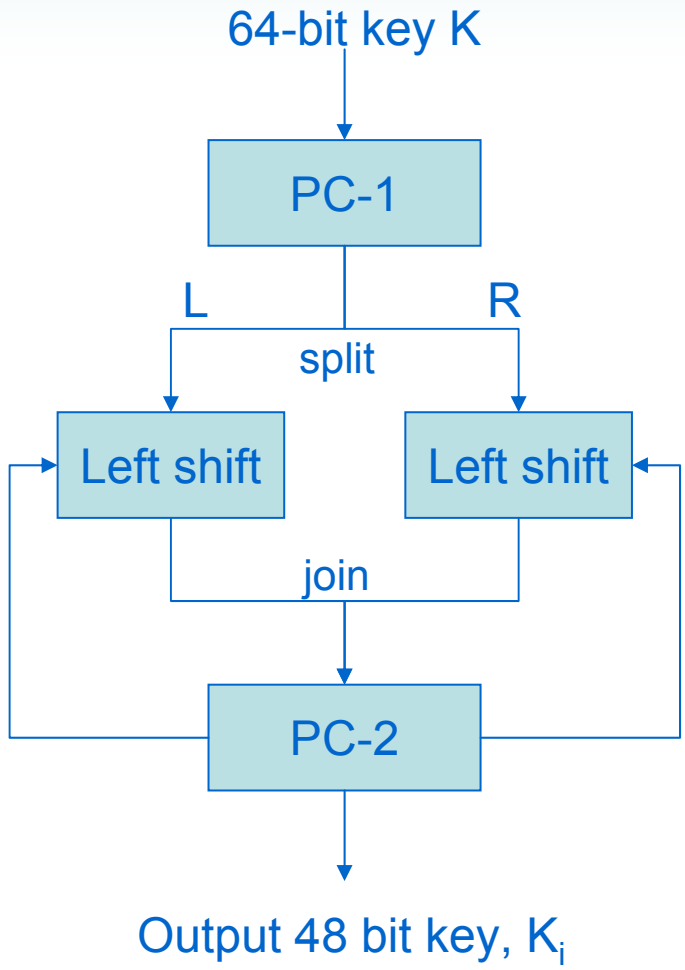
DES $$ciphertext = IP^{-1}\left(f_{K_{16}}\left(SW\left(f_{K_{15}}\left(SW\left(\ldots\left(SW\left(f_{K_1}\left(IP\left(plaintext\right)\right)\right)\right)\right)\right)\right)\right)\right)$$

# Generic DES Algorithm



# DES Key Generation

- Map 64-bit original key (K) to 56-bits (ignore every 8th bit which is used for parity check)
- Re-arrange bits according to PC-1, and split into two halves, L and R
  - E.g. bit 1 of L will be the 57th bit of K, bit 1 of R will be 63rd bit of K
- Apply left shift either 1 or 2 positions (according to round)
- Join and apply PC-2 to produce output key, $K_i$
- Put Split $K_i$ and use as input to left shifts for next round



# DES Key Generation

**(a) Input Key**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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)**

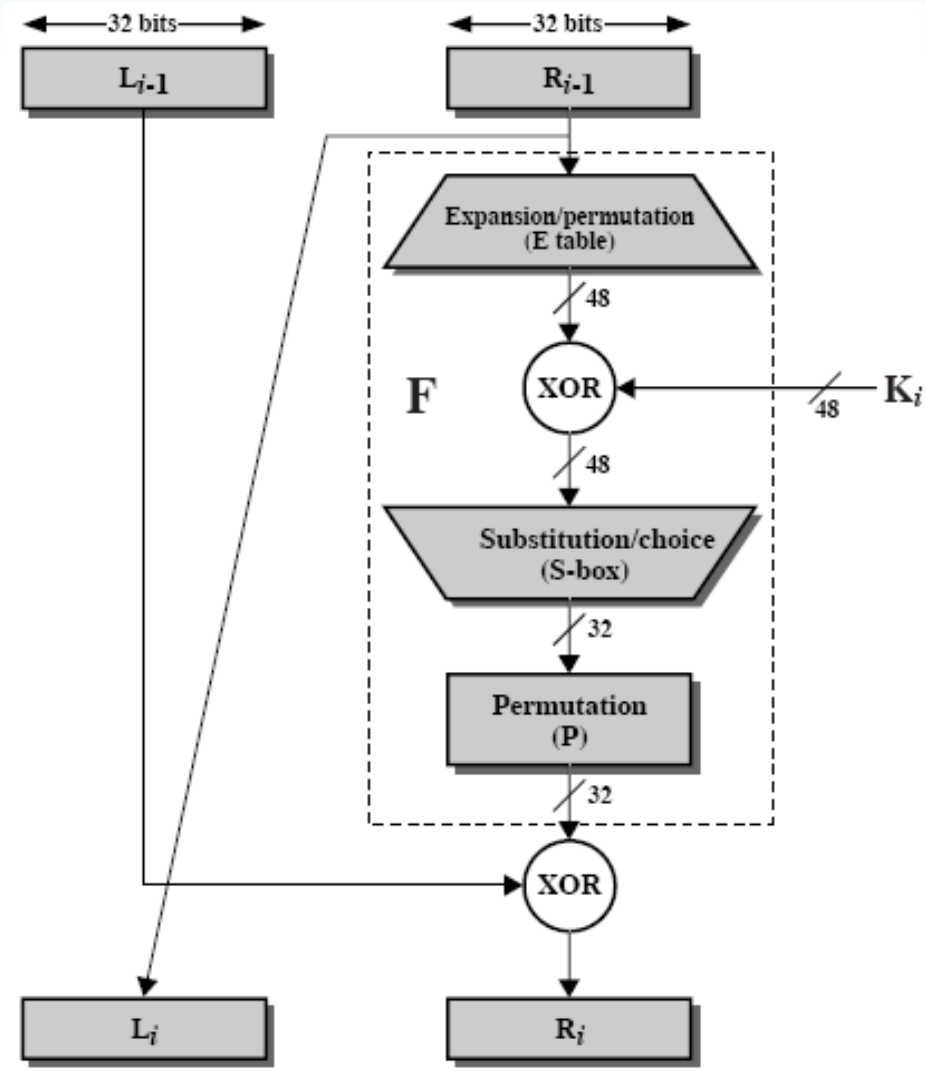
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 14 | 17 | 11 | 24 | 1 | 5 | 3 | 28 |
| 15 | 6 | 21 | 10 | 23 | 19 | 12 | 4 |
| 26 | 8 | 16 | 7 | 27 | 20 | 13 | 2 |
| 41 | 52 | 31 | 37 | 47 | 55 | 30 | 40 |
| 51 | 45 | 33 | 48 | 44 | 49 | 39 | 56 |
| 34 | 53 | 46 | 42 | 50 | 36 | 29 | 32 |

**(d) Schedule of Left Shifts**

| Round number | 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 |

# Single Round of DES

- Split 56-bit block into L and R
- Expand R to 48 bits and permute (re-arrange) using E
- XOR E with $K_i$
- Split into 8 x 6-bit values and input into 8 S-Boxes
- Join S-Box outputs and permute using P, then XOR with L to produce output of R'
- L' = R
- Move to next round using L' and R' as input



# DES Permutations

**(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 |

**(b) 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 |

**(c) 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 |

**(d) 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 |

# DES S-Boxes 1-4

$S_1$

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

# DES S-Boxes 5-8

$S_5$

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

# Some Design Issues with DES

- Avalanche Effect
  - Aim: small change in plaintext or key produces large change in ciphertext
  - DES achieves this quite well
  - Table shows change in cipher text bits after each round
  - Plaintext differ by 1 bit
  - Keys differ by 1 bit
  - Both cases the output ciphertext differs by more than 30 bits

| (a) Change in Plaintext | | (b) Change in Key | |
|---|---|---|---|
| Round | Number of bits that differ | Round | Number of bits that differ |
| 0 | 1 | 0 | 0 |
| 1 | 6 | 1 | 2 |
| 2 | 21 | 2 | 14 |
| 3 | 35 | 3 | 28 |
| 4 | 39 | 4 | 32 |
| 5 | 34 | 5 | 30 |
| 6 | 32 | 6 | 32 |
| 7 | 31 | 7 | 35 |
| 8 | 29 | 8 | 34 |
| 9 | 42 | 9 | 40 |
| 10 | 44 | 10 | 38 |
| 11 | 32 | 11 | 31 |
| 12 | 30 | 12 | 33 |
| 13 | 30 | 13 | 28 |
| 14 | 26 | 14 | 26 |
| 15 | 29 | 15 | 34 |
| 16 | 34 | 16 | 35 |

# Some Design Issues with DES

- Initial/Final Permutation (IP and $IP^{-1}$) – why?
  - The initial permutations have no security value
  - The original algorithm that DES was based on did not have these permutations
  - Possible reason for IP: make DES less efficient to implement in software (no longer relevant; software implementations now feasible)
- 56-bit keys,
  - 1 decryption per µs: 1142 years
  - In 1977, estimated cost $US20m to build machine to break it in 10 hours
  - In 1988, EFF built machine for $250k that broke DES in 3 days
  - Remember: key searches require ability to identify the plaintext (not so easy in some cases)
  - Triple DES (3DES) uses 128-bit keys

# Some Design Issues with DES

- S-Boxes: why are they designed as they are?
  - How did designers chose the S-Boxes?
    - DES was a closed design process in 1970's. Why? One reason is so designers did not divulge information about known attacks
  - Experience has shown that it looks like a good design (no one has found flaws)
    - Experiments have shown that simple changes (eg. Swap S-Box 3 with S-Box 7) and DES becomes much weaker
  - A lot of research has been conducted on S-Box design and there are some general principles that should be followed