Symmetric Key Block Encryption Cipher

Data Encryption Standard (DES)

Jibi Abraham

COEP TECHNOLOGICAL UNIVERSITY

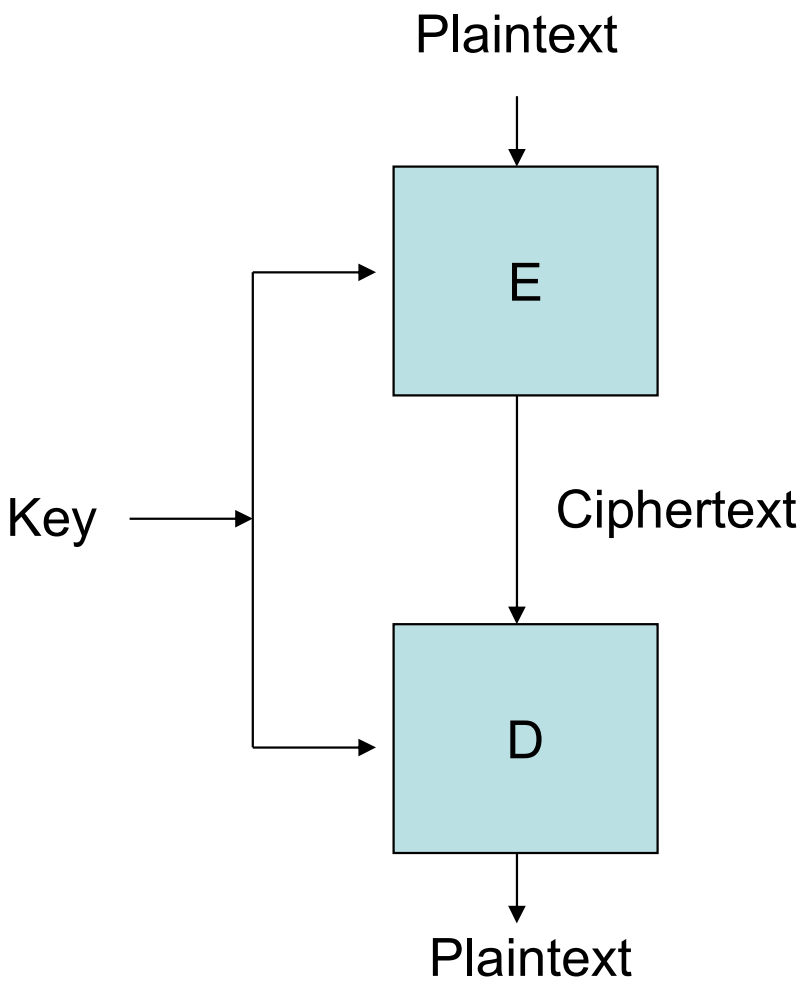
Shivajinagar, Pune-411 005

(A Unitary Technological University of Govt. of Maharashtra)

Cipher Principles

- **Stream Ciphers and Block Ciphers**
- Stream cipher: one bit or byte at a time
- Block cipher: a large block, typically 64 or 128 bits, at a time
  - Principle behind: Large block thwarts statistical attacks
  - Block ciphers work on a block at a time, which is some number of bits. All of these bits have to be available before they can be processed
  - Broader range of applications than stream ciphers

Block Ciphers



- DES
- 3DES
- AES
- Twofish
- Blowfish
- Serpent
- IDEA

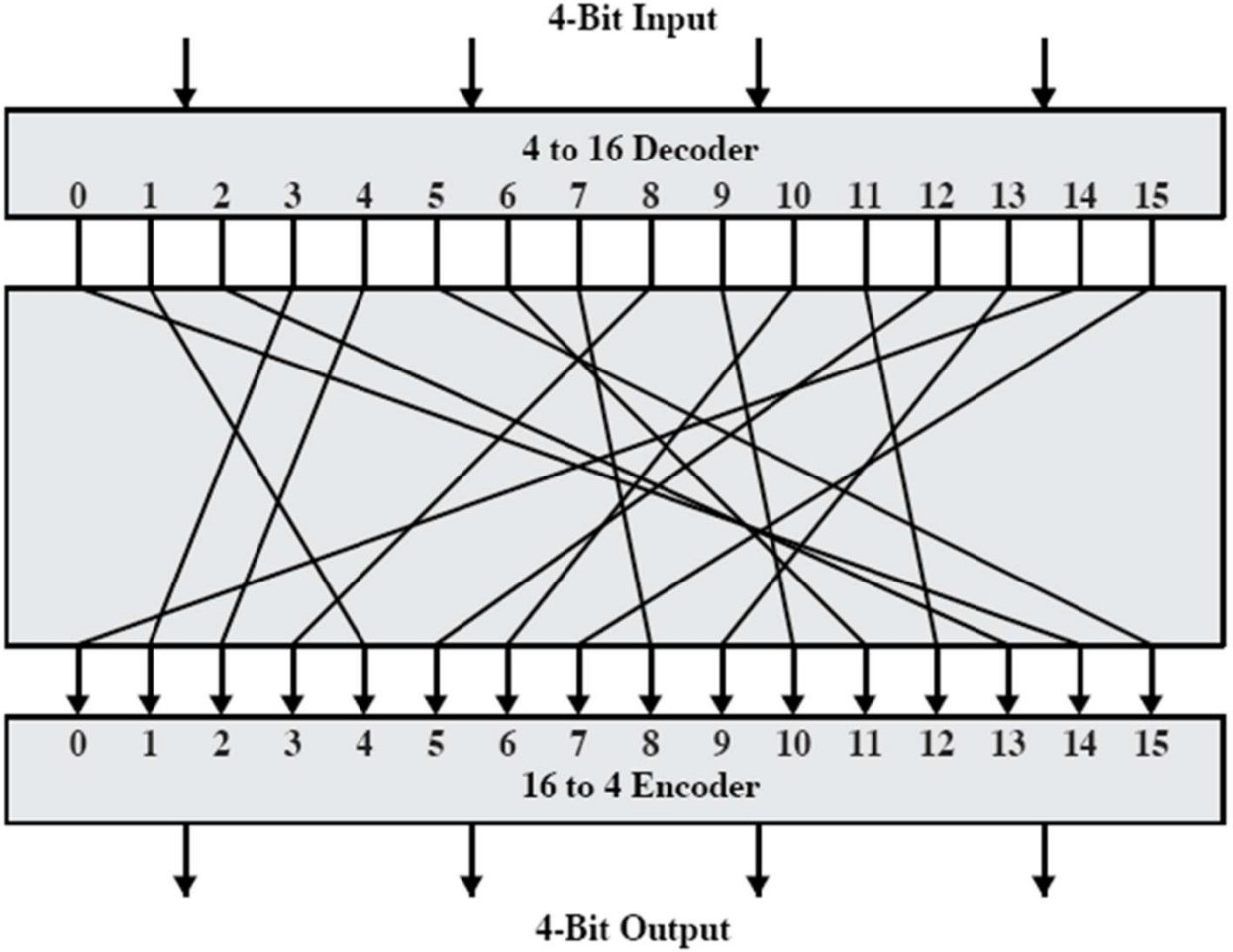
Block Cipher Principles

- **Motivation for the Feistel Cipher Structure**
- The most general form of block cipher
  - For n-bit block, consider the transformation, $\mathbb{F}$: $2^n \rightarrow 2^n$
  - $\mathbb{F}$ must be reversible, i.e., 1-1 correspondence
  - (Ideal block cipher) $2^n!$ Mappings, each should produce a unique cipher block

Reversible Mapping	
Plaintext	Ciphertext
00	**11**
01	**10**
10	**00**
11	**01**

Irreversible Mapping	
Plaintext	Ciphertext
00	**11**
01	**10**
10	**01**
11	**01**

Ideal Block Cipher

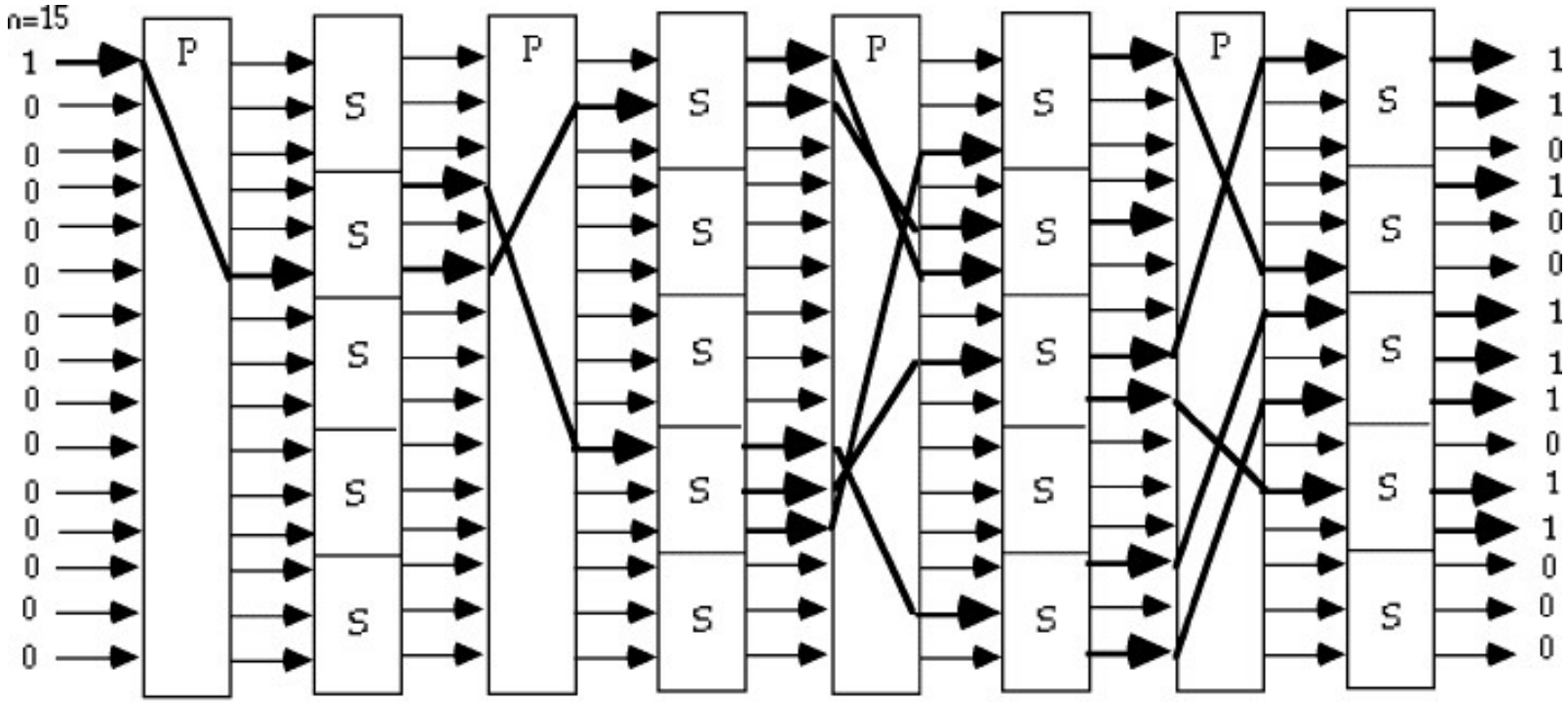


Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

- Mapping F itself is a key $\Rightarrow$ key size = $O(n \times 2^n)$ bits
- 64-bit block $\Rightarrow$ $64 \times 2^{64} \approx 2^{70} \approx 10^{21}$ bits key
- $\Rightarrow$ Not practical

Shannon's Substitution-Permutation (S-P) Networks

- In 1949, Claude Shannon introduced the idea of substitution-permutation (S-P) networks which form the basis of modern block ciphers



Data Encryption Standard (DES)

- Financial companies found the need for a cryptographic algorithm that would have the blessing of the US government (National Security Agency (NSA))
- In 1973, NBS (NIST) issued a public request for proposals for a national cipher standard, which must be
  - Secure, Public, Completely specified, Easy to understand, Available to all users, Economic and efficient in hardware, Able to be validated, Exportable
- IBM submitted LUCIFER (Feistel) (which was redesigned to become the DES)

DES

- In 1977, adopted by NBS (NIST) as DES (Data Encryption Standard, Federal Information Processing Standard 46 (FIPS PUB 46))
- DES became a federal standard in November 1976
  - NBS (NIST) hardware standard in January 77
  - ANSI X3.92-1981 (hardware + software)
  - ANSI X3.106-1983 (modes of operation)
  - Australia AS2805.5-1985
- Used in most EFT and EFTPOS from banking industry
  - It was reconfirmed as a standard for 5 years twice
  - Currently 3DES is recommended

Simplified DES

- Input (plaintext) block: 8-bits
- Output (ciphertext) block: 8-bits
- Key: 10-bits
- Rounds: 2
- Round keys generated using permutations and left shifts
- Encryption: initial permutation, round function, switch halves
- Decryption: Same as encryption, except round keys used in opposite order

Comparing DES and S-DES

- S-DES
  - 8-bit blocks
  - 10-bit key: 2 x 8-bit round keys
  - IP: 8-bits
  - F operates on 4 bits
  - 2 S-Boxes
  - 2 rounds
- DES
  - 64-bit blocks
  - 56-bit key: 16 x 48-bit round keys
  - IP: 64 bits
  - F operates on 32 bits
  - 8 S-Boxes
  - 16 rounds

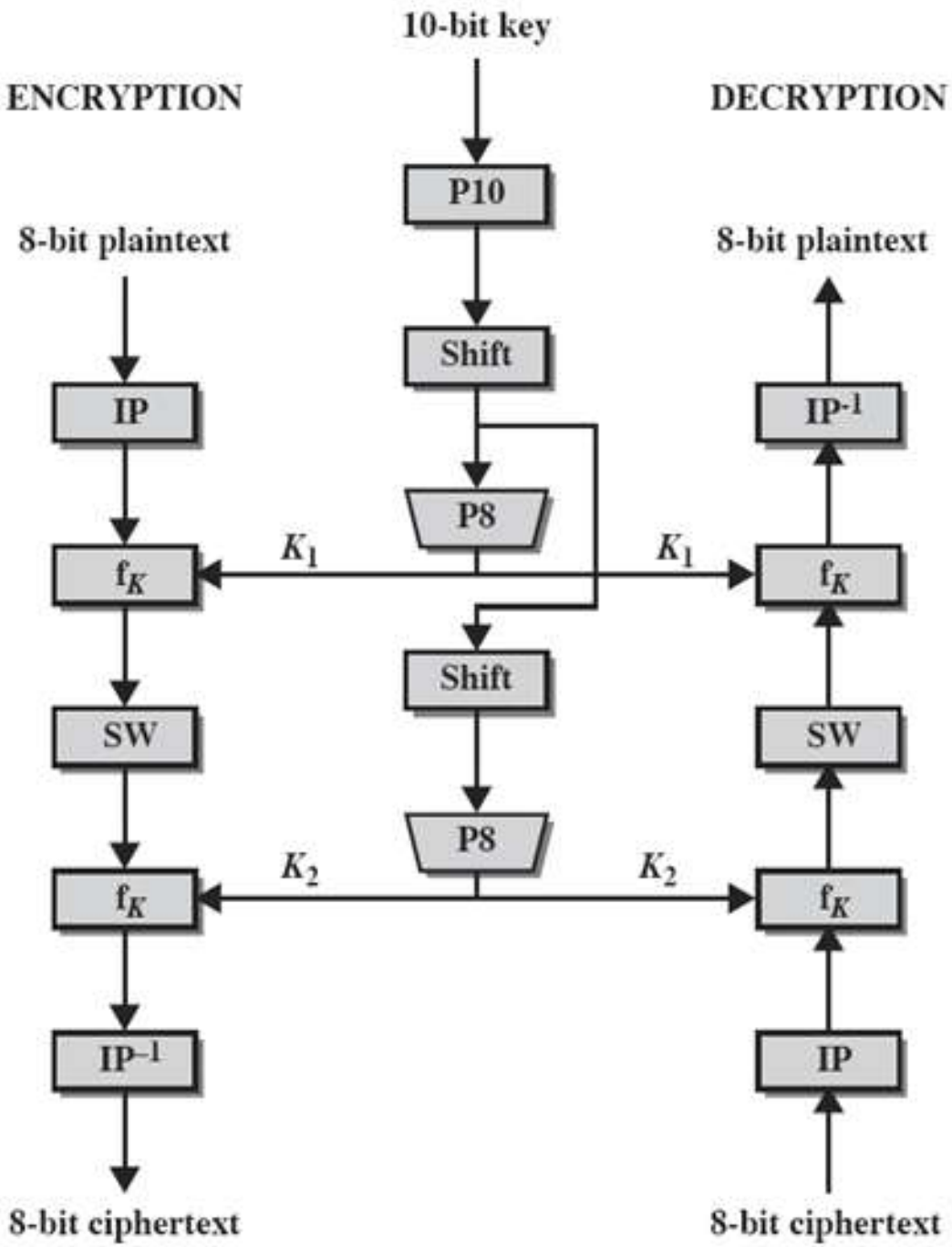
S-DES encryption:

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

DES encryption:

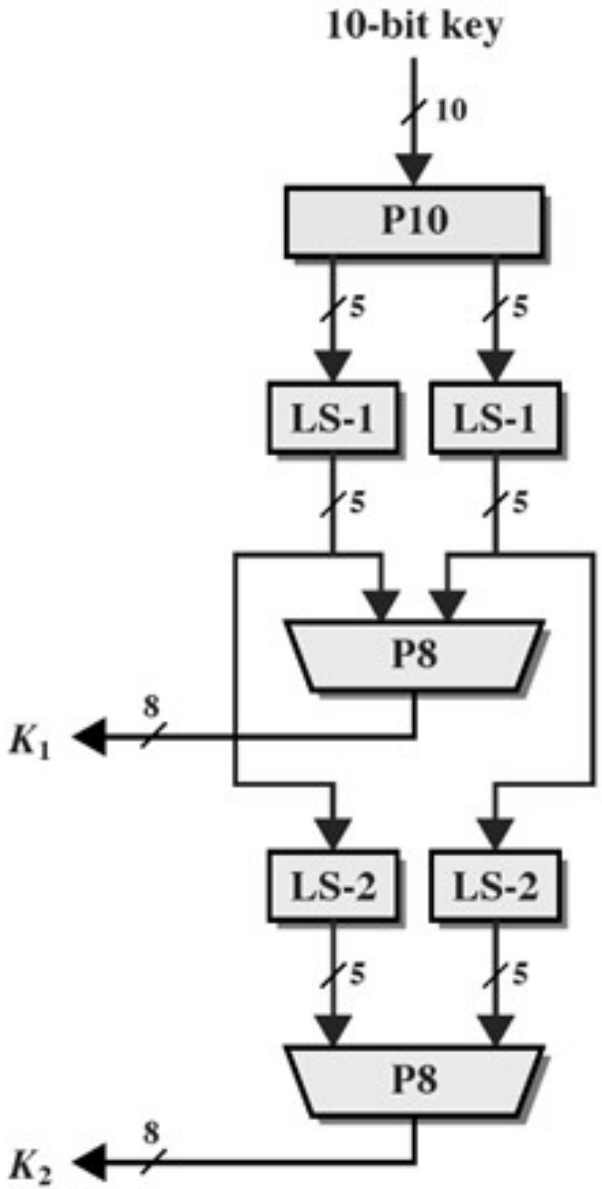
$$ciphertext = IP^{-1}\left(f_{K_{16}}\left(SW\left(f_{K_{15}}\left(SW\left(\ldots\left(f_{K_1}\left(IP\left(plaintext\right)\right)\right)\right)\right)\right)\right)\right)$$

S-DES Algorithm

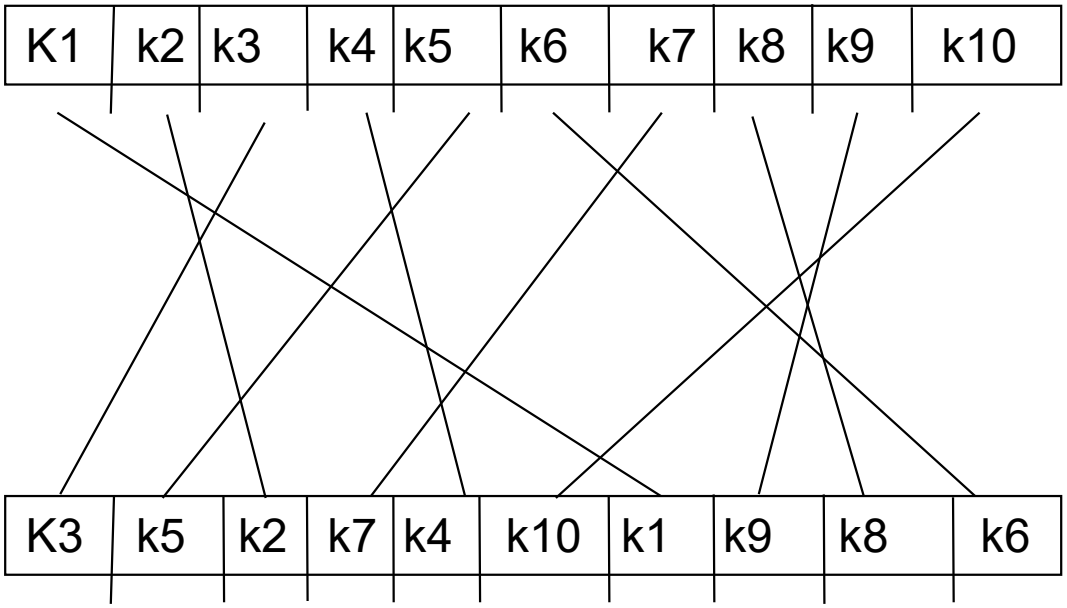


Simple DES-Key Generation

- Initial permutation P10
- Divide in left and right parts
- Left shift and Merge
- An 8 bits permutation, resulting a 8 bits K1
- Divide in left and right parts
- Double left shift and Merge
- An 8 bits permutation, resulting a 8 bits K2



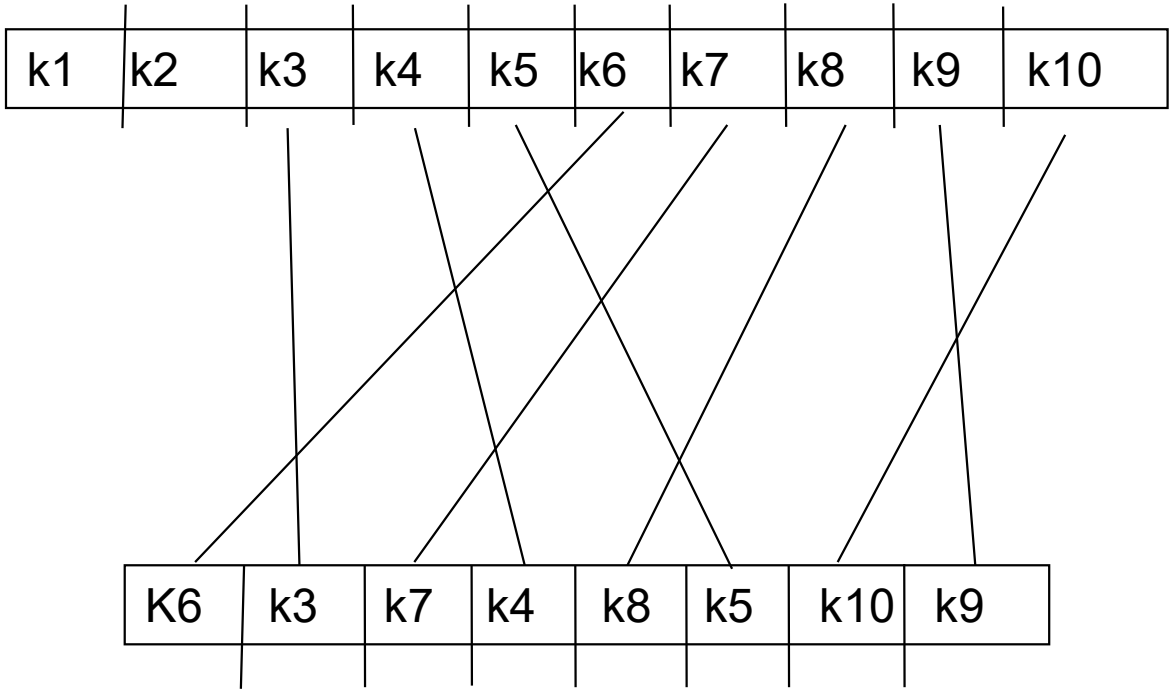
P10 Permutation



P10 = { 3, 5, 2, 7, 4, 10, 1, 9, 8, 6}

P8 Permutation

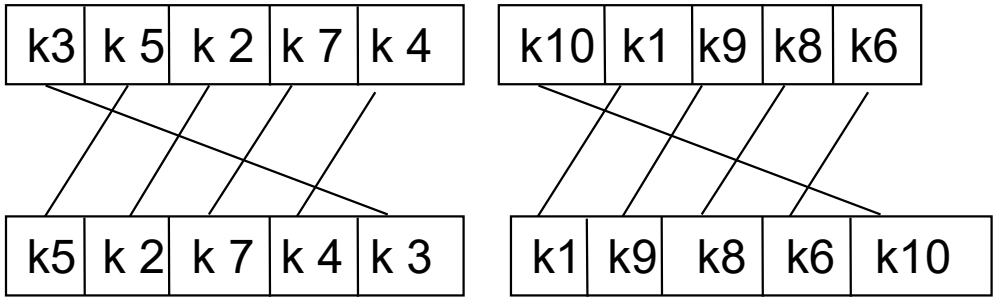
Permute 10 into 8



P8 = { 6, 3, 7, 4, 8, 5, 10, 9}

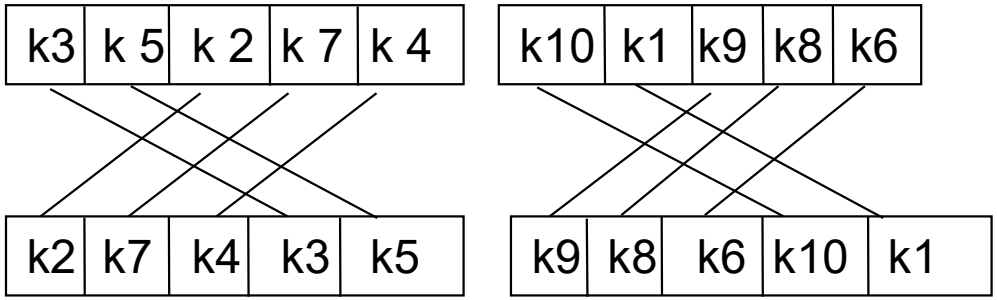
LS-1

Left circular shift 1 each 5-bit group



LS-2

Left circular shift 2 each 5 bit group



S-DES Key generation

P10 = { 3, 5, 2, 7, 4, 10, 1, 9, 8, 6}

K = 1 0 1 0 0 0 0 0 1 0

After P10 = 1 0 0 0 0 0 1 1 0 0

Split: 10000 01100

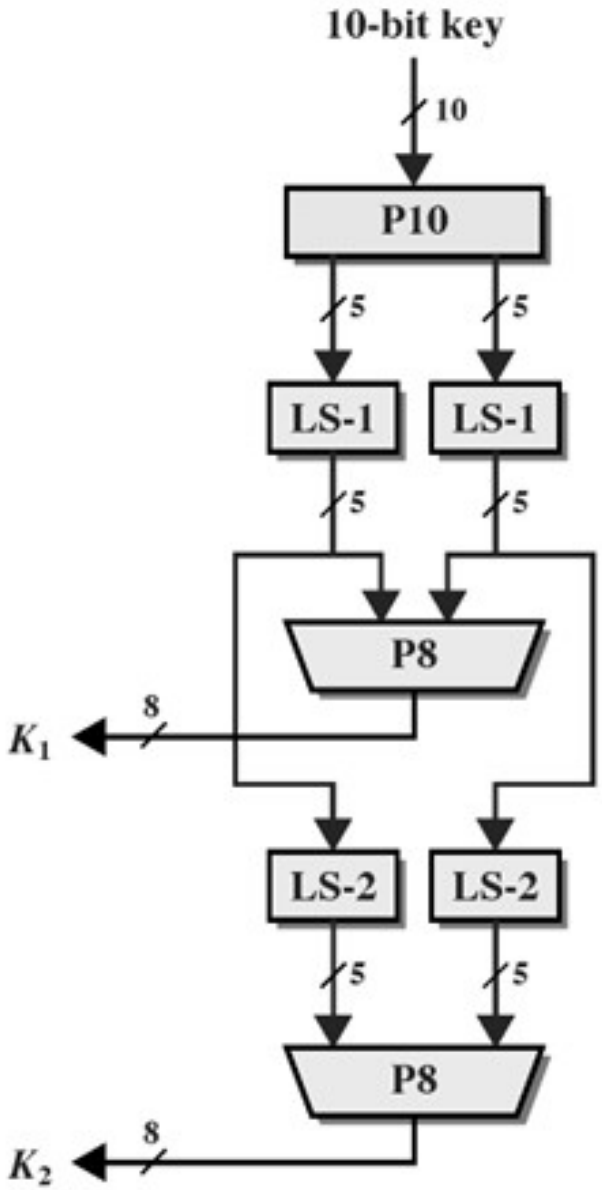
After LS-1 = 0 0 0 0 1 1 1 0 0 0

P8 = { 6, 3, 7, 4, 8, 5, 10, 9}

After P8 (K1) = 1 0 1 0 0 1 0 0

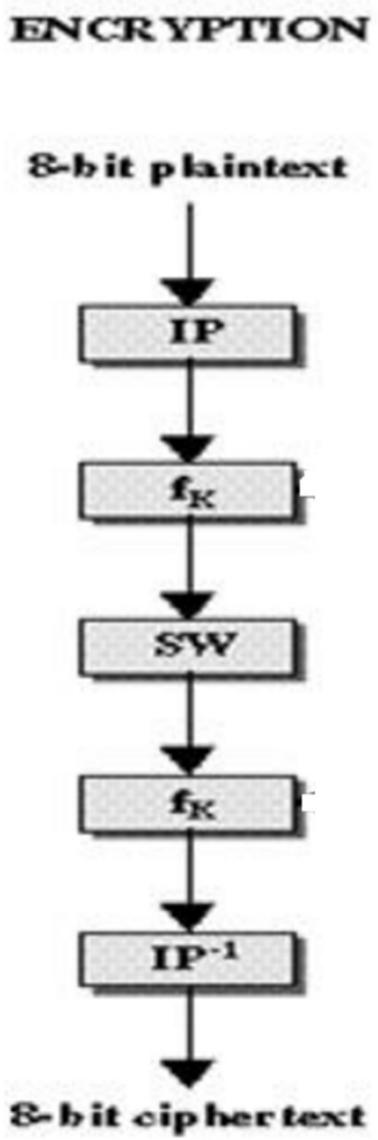
After LS-2 = 0 0 1 0 0 0 0 0 1 1

After P8 (K2)= 0 1 0 0 0 0 1 1

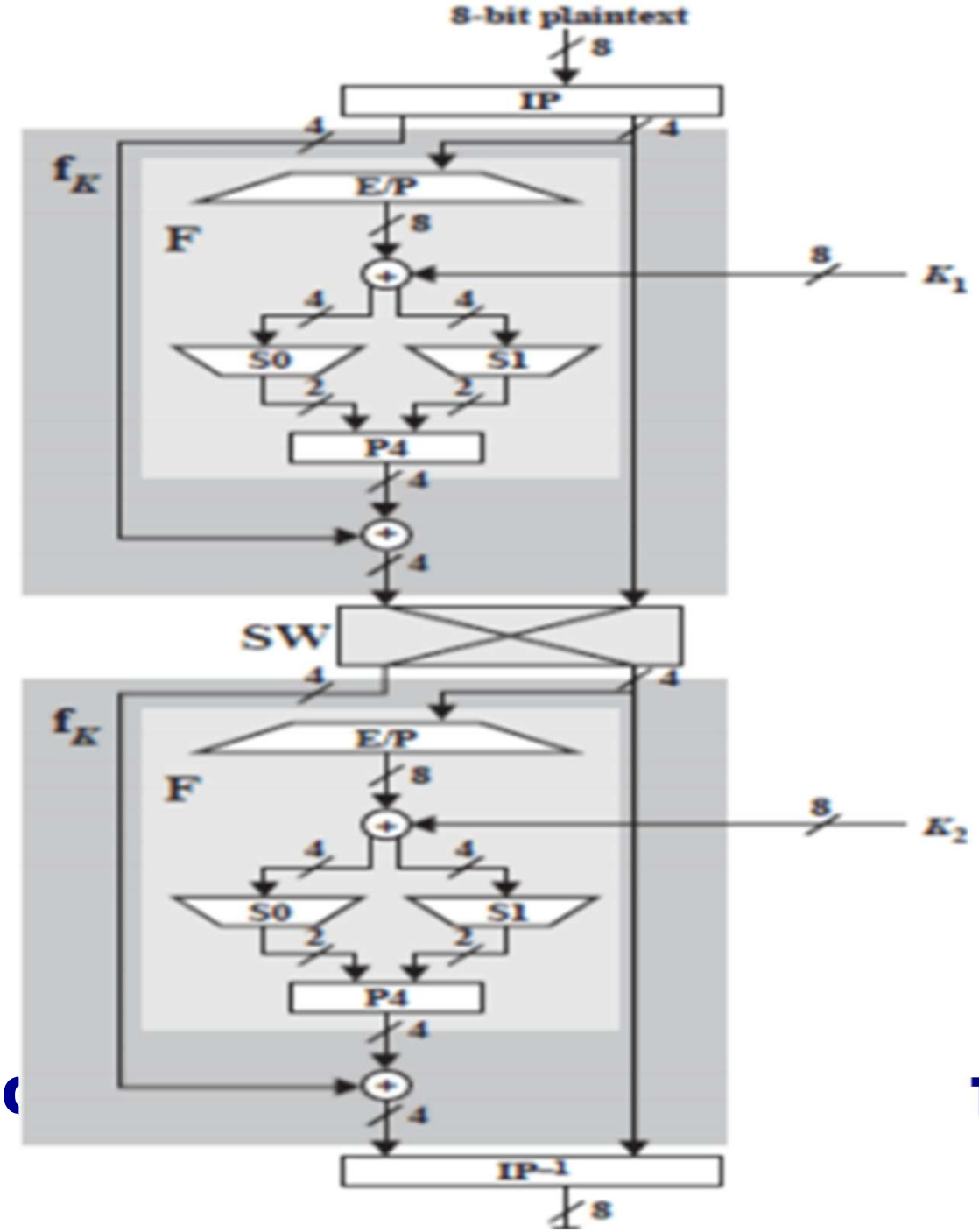


Simple DES - Encryption

- The encryption process is :
  - Initial Permutation
  - Function f_k $$f_K(L,R) = (L \oplus F(R,SK),R)$$
  - Switch of the key halves
  - Function f_k $$f_K(L,R) = (L \oplus F(R,SK),R)$$
  - Final Permutation (inverse of initial permutation)

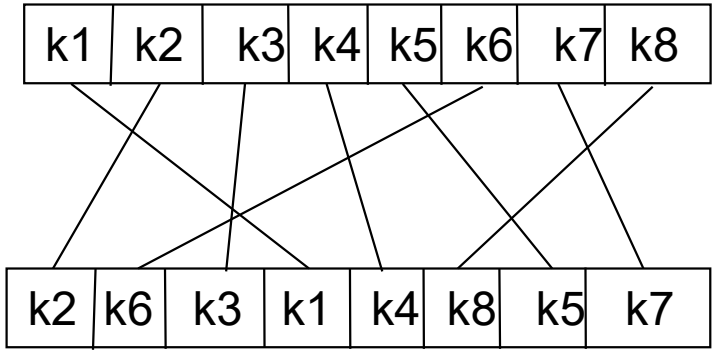


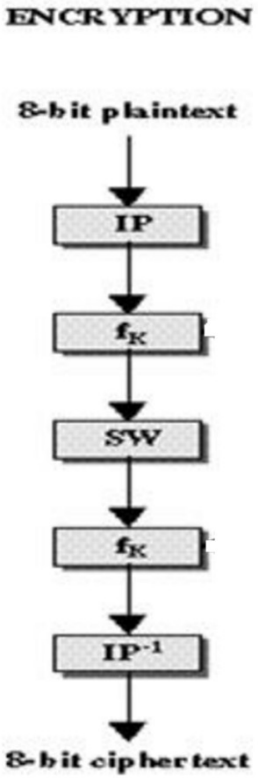
DES - Encryption



Initial Permutation (IP)

Move the bits of the original character around a little…



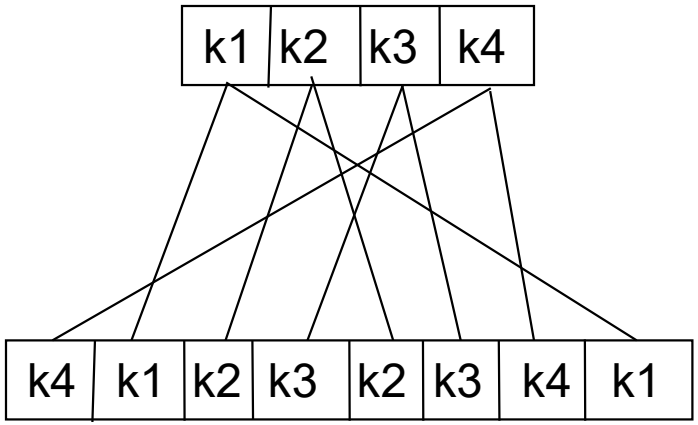


IP = { 2, 6, 3, 1, 4, 8, 5, 7}

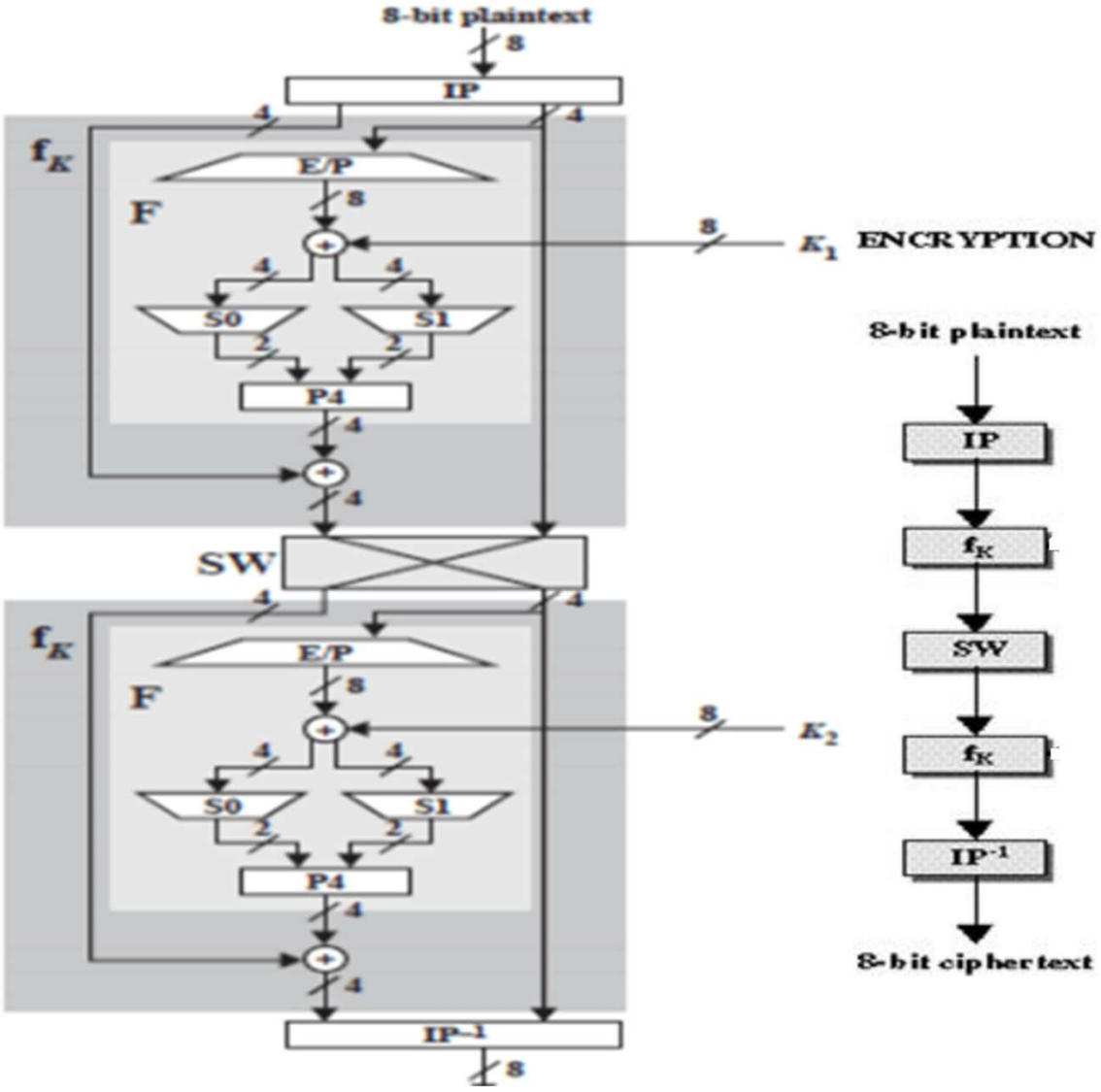
IP^{-1}= { 4, 1, 3, 5, 7, 2, 8, 6}

Expansion/Permutation (E/P)

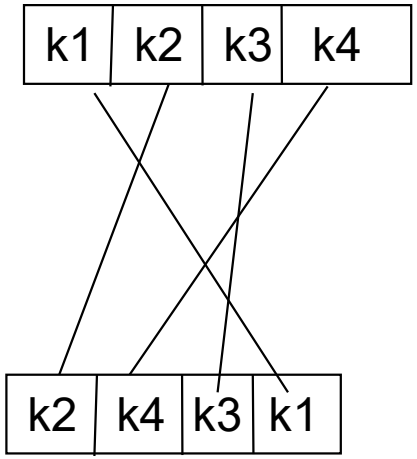
In f_k, expand 4 bits into 8 and permutate them…

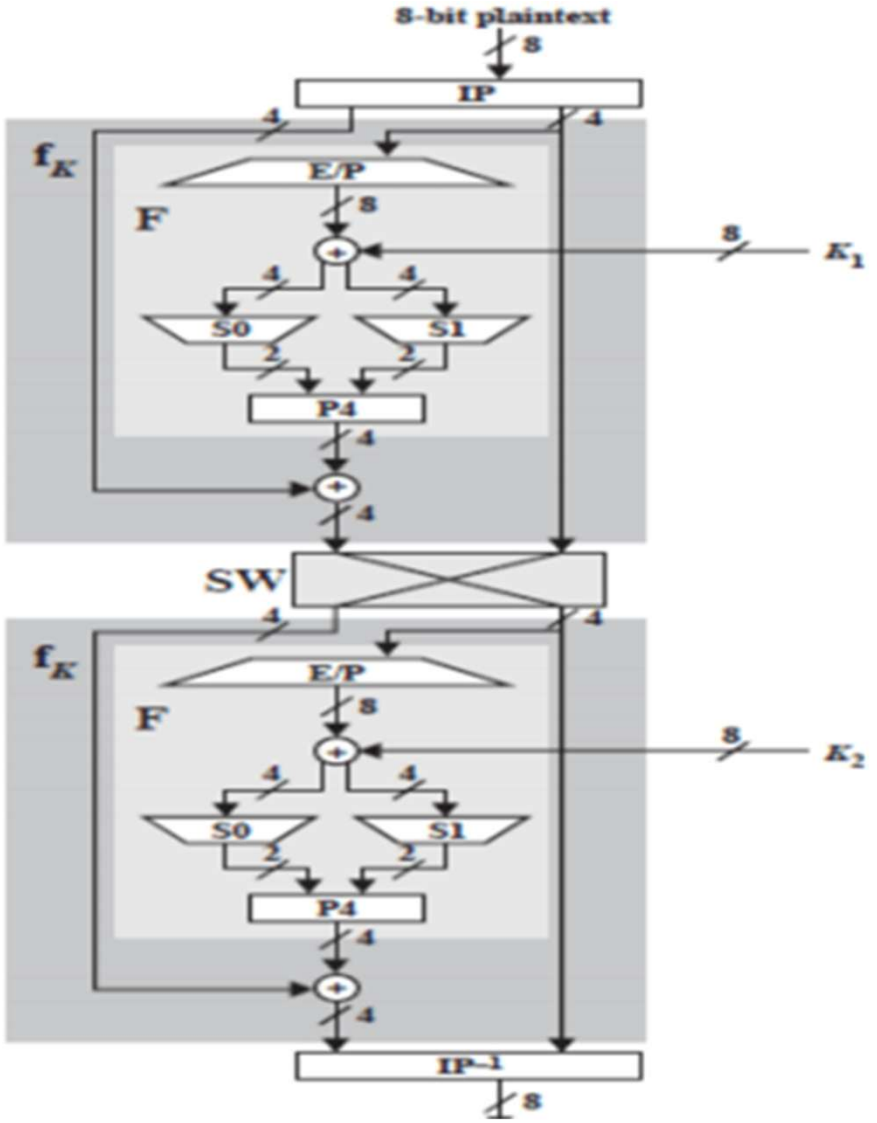


$$\begin{pmatrix} 4 & 1 & 2 & 3 & 2 & 3 & 4 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$$



P4 Permutation





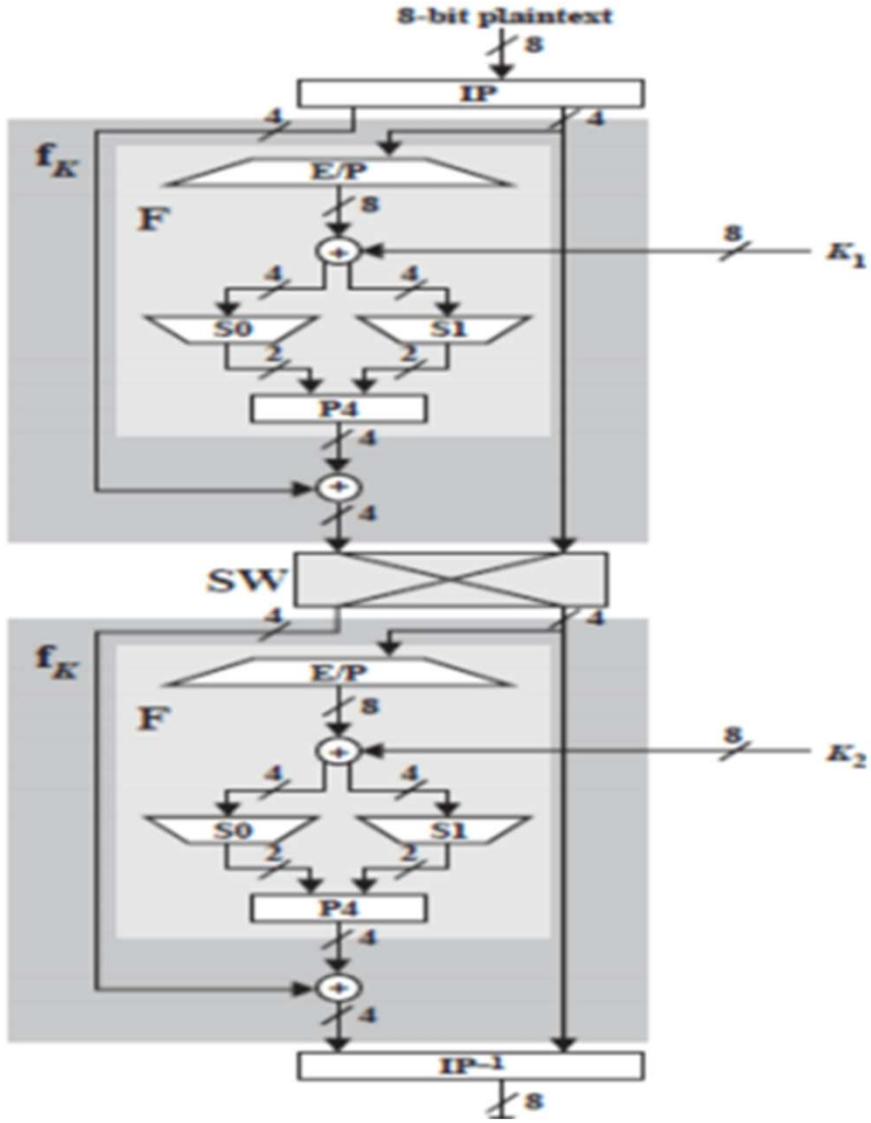
Substitution Boxes

S0

1	0	3	2
3	2	1	0
0	2	1	3
3	1	3	2

S1

0	1	2	3
2	0	1	3
3	0	1	0
2	1	0	3



S-DES Encryption Details

Plaintext - 1 0 0 1 0 1 1 1

IP = { 2, 6, 3, 1, 4, 8, 5, 7 }

After IP = 0 1 0 1 1 1 0 1

L = 0 1 0 1 and R = 1 1 0 1

EP = { 4, 1, 2, 3, 2, 3, 4, 1}

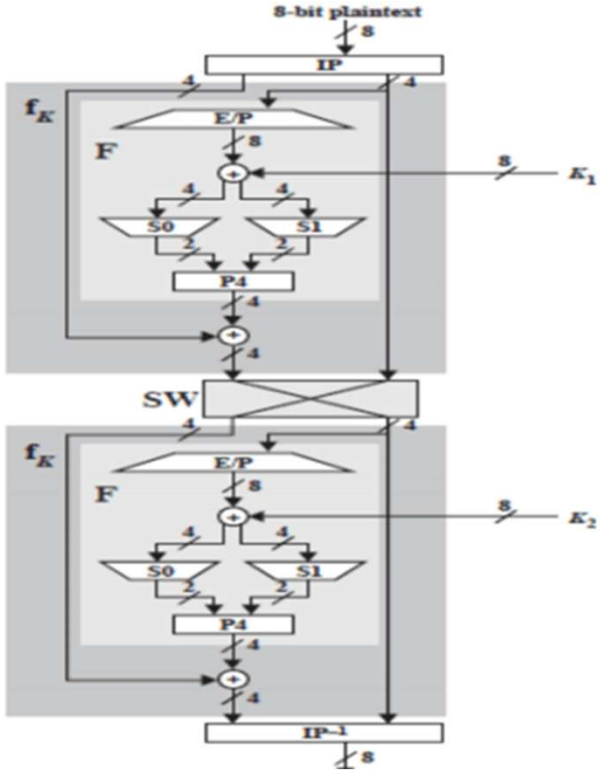
After EP = 1 1 1 0 1 0 1 1

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

Let K1 - 1 0 1 0 0 1 0 0

XOR with K1 = 0 1 0 0 1 1 1 1

L = 0 1 0 0 and R = 1 1 1 1



S-Box

- S-DES (and DES) perform substitutions using S-Boxes
- S-Box considered as a matrix: input used to select row/column; selected element is output
- 4-bit input: bit1; bit2; bit3; bit4
  - bit1, bit4 specify row (0, 1, 2 or 3 in decimal)
  - Bit2, bit3 specify column
  - 2-bit output

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

S-DES Encryption

L = 0 1 0 0 and R = 1 1 1 1

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

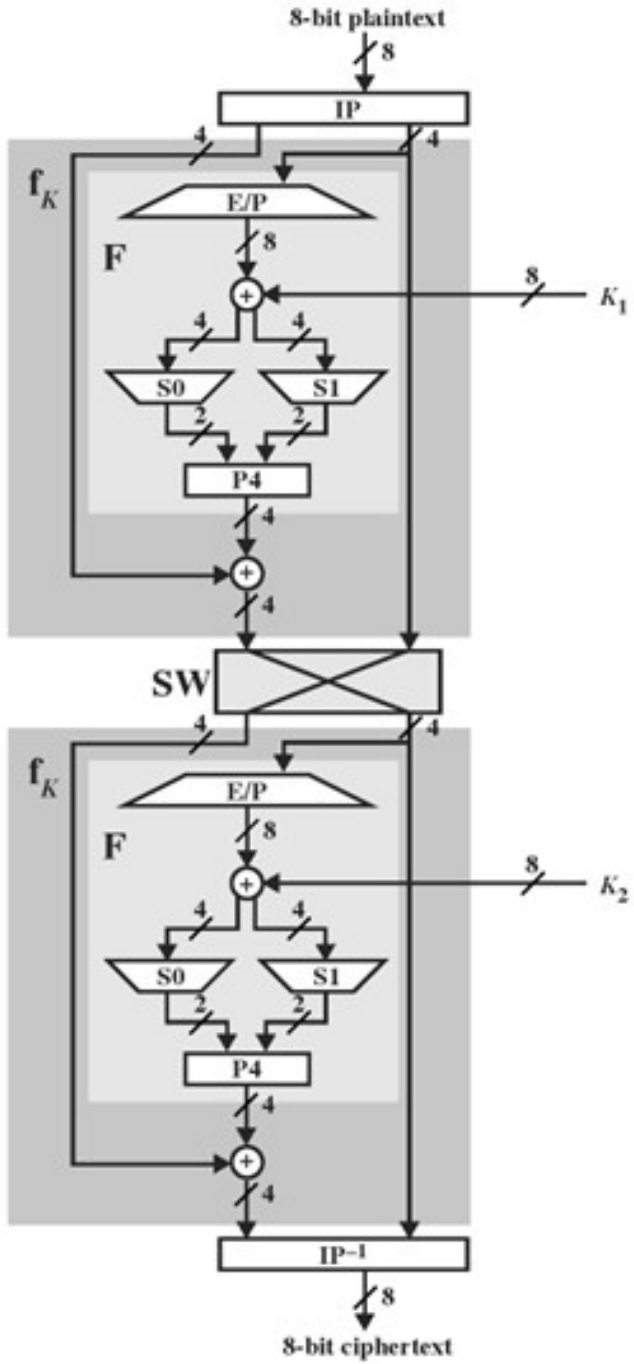
For L = 0 1 0 0
Row(1st and 4th) = 00(0),
column(2nd and 3rd) = 10(2)
S0 = 3(11)

For R = 1 1 1 1
row = 11(3), column = 11(3) S1 = 3(11)

After S0 and S1 = 1 1 1 1

P4 = { 2, 4, 3, 1}

After P4 = 1 1 1 1



S-DES Encryption

Plaintext- L = 0 1 0 1 and R = 1 1 0 1

After P4 L = 0 1 0 1 and R = 1 1 1 1

(0 1 0 1) XOR (1 1 1 1) = 1 0 1 0

After Swap L= 1 1 0 1 and R = 1 0 1 0

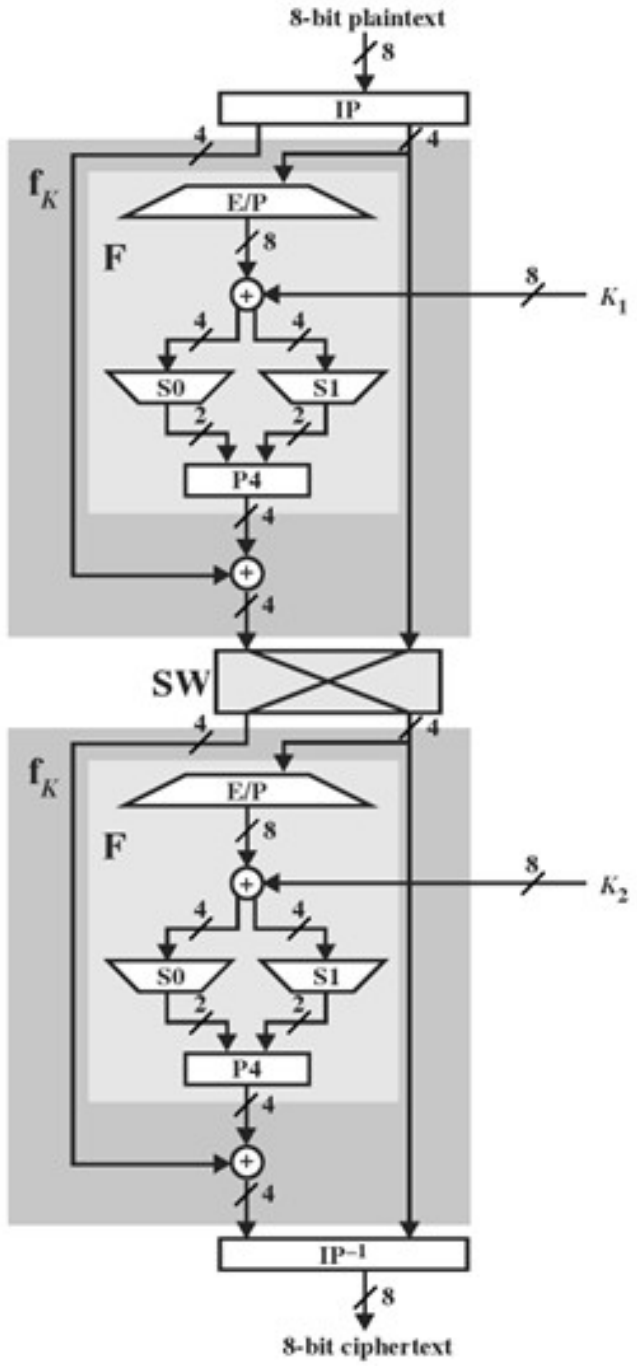
K2 - 0 1 0 0 0 0 1 1

After XOR = 0 0 1 0

Input to IP^{-1} = 0 0 1 0 1 0 1 0

IP^{-1} = { 4, 1, 3, 5, 7, 2, 8, 6}

Ciphertext = **0 0 1 1 1 0 0 0**



S-DES Summary

- S-DES expressed as functions:

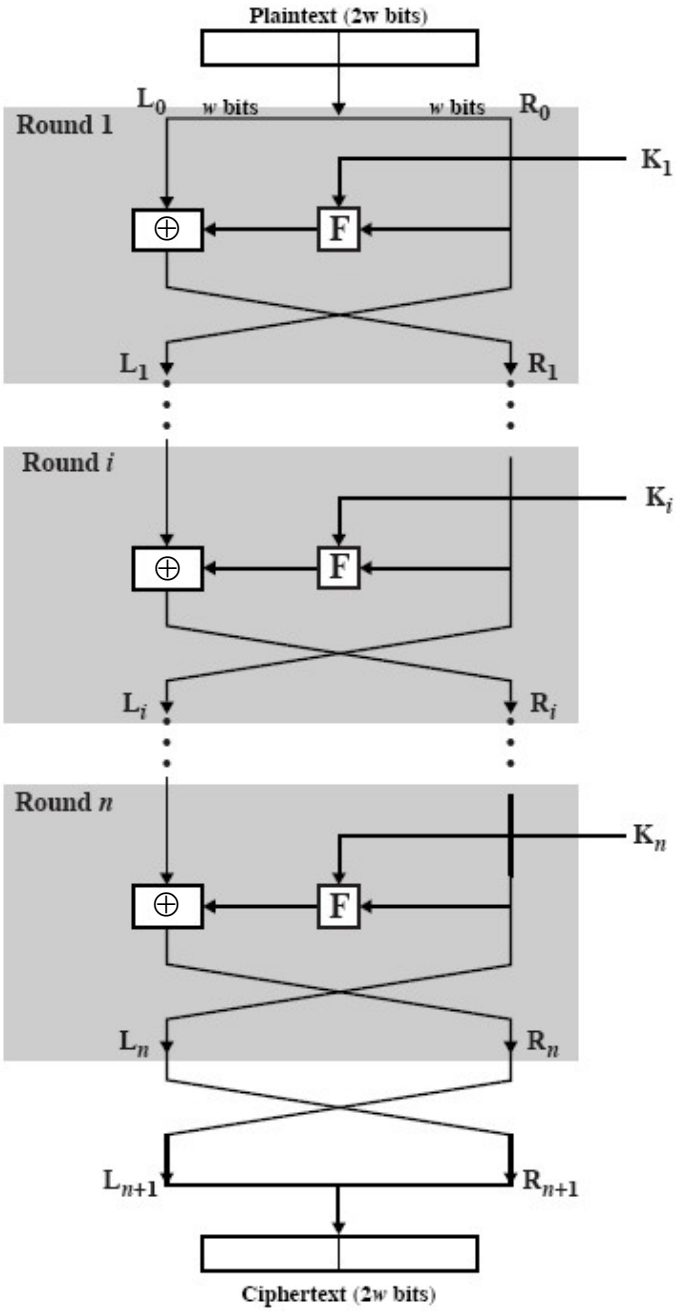
$$ciphertext = IP^{-1}\left(f_{K_2}\left(SW\left(f_{K_1}\left(IP\left(plaintext\right)\right)\right)\right)\right)$$

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

- Security of S-DES:
  - 10-bit key, 1024 keys: brute force easy
  - If know plaintext and corresponding ciphertext, can we determine key? Very hard

Motivation for Feistel Cipher Structure

- S-P network: a special form of substitution-transposition product cipher
- Product cipher
  - Two or more simple ciphers are performed in sequence in such a way that the final result or product is cryptographically stronger than any of the component ciphers
- Feistel cipher
  - In 1970's, Horst Feistel (IBM T.J. Watson Research Labs) invented a suitable structure which adapted Shannon's S-P network
  - Encryption and decryption use the same structure



Diffusion and Confusion

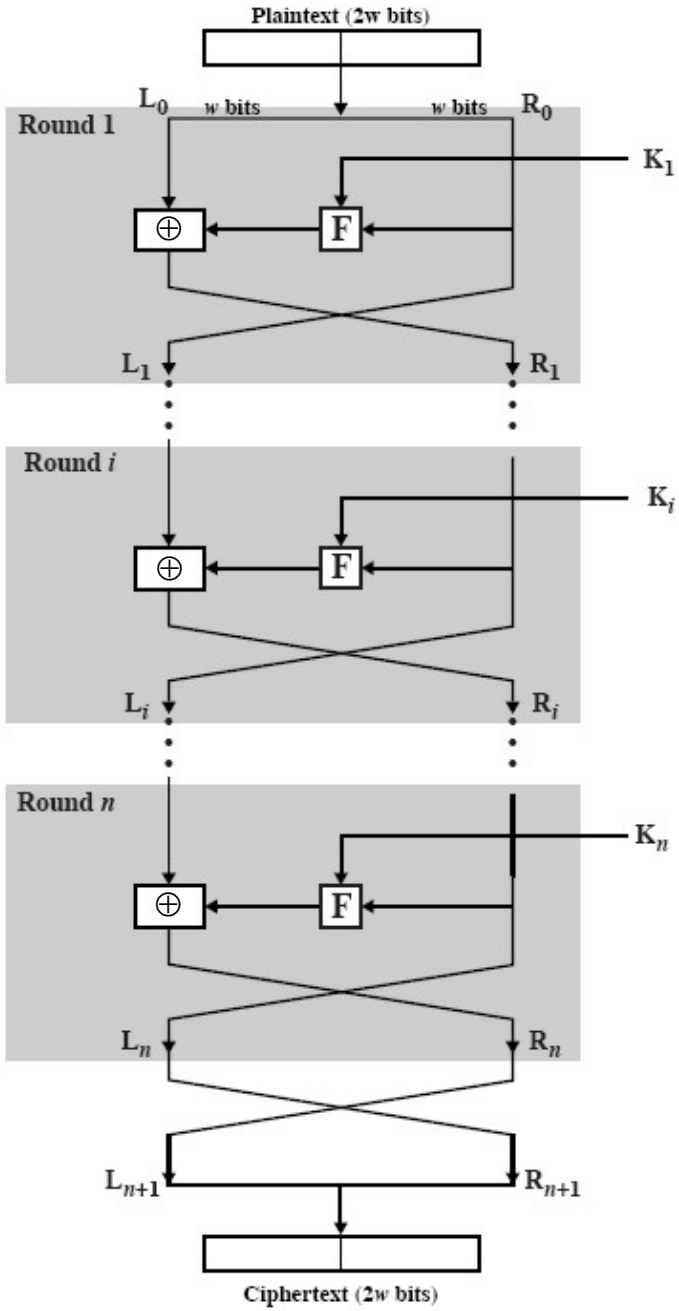
- Apply *diffusion* and *confusion* operations to thwart cryptanalysis based on statistical analysis
- Diffusion
  - Dissipate statistical structure of the plaintext into long-range statistics of the ciphertext
  - Spread the statistics over a range of bits, i.e., let each part of the plaintext affect a large part of the ciphertext, thus making the statistical relationship as difficult as possible
  - Thwart frequency analysis
  - Can be achieved by repeatedly performing some permutation followed by applying a function to that permutation
- Confusion
  - Make statistical relationship between the ciphertext and key as difficult as possible
  - Thwart attempts to discover the key
  - Can be achieved by using a complex, non-linear, substitution operation (S-box)

Feistel Cipher Structure

- Input block is partitioned into two halves, L_{i-1} and R_{i-1}
- In round i,

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

- Substitution followed by permutation
- Multiple rounds
- An Implementation of Shannon's S-P network (SPN) concept

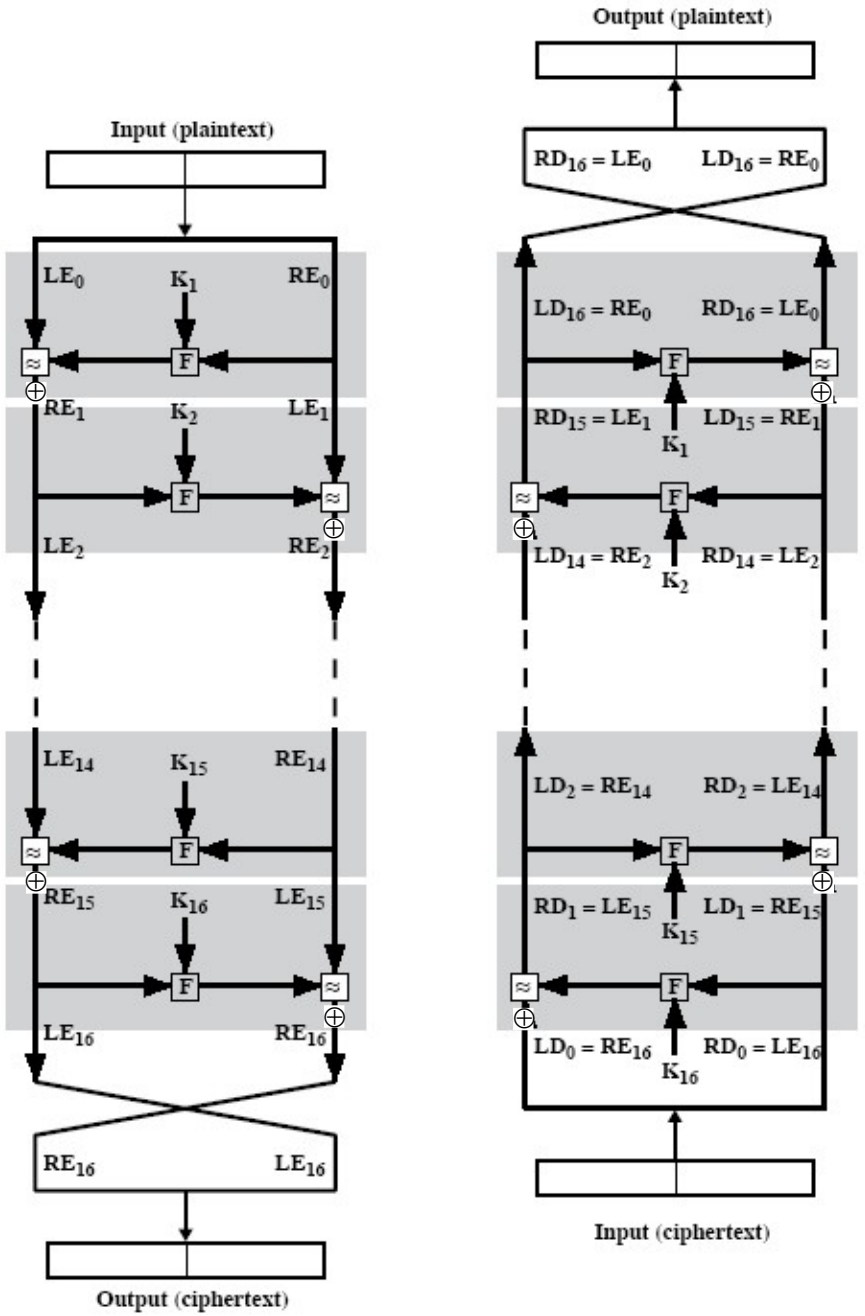


Feistel Cipher Design Elements

- Block size
  - Increasing size improves security, but slows cipher
- Key size
  - Increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- Number of rounds
  - Increasing number improves security, but slows cipher
- Subkey generation algorithm
  - Greater complexity can make analysis harder, but slows cipher
- Round function
  - Greater complexity can make analysis harder, but slows cipher
- To be fast in software implementation of encryption/decryption

Feistel Decryption Algorithm

- Essentially same as the encryption
- Same algorithm but use the subkeys K_i in reverse order
- F need not be a reversible function
- Satisfy $D_K(E_K(P)) = P$



Feistel Decryption Algorithm

To show $LD_i \,||\, RD_i = RE_{16-i} \,||\, LE_{16-i}$

Proof:

$LE_{16} = RE_{15}$

$RE_{16} = LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16})$

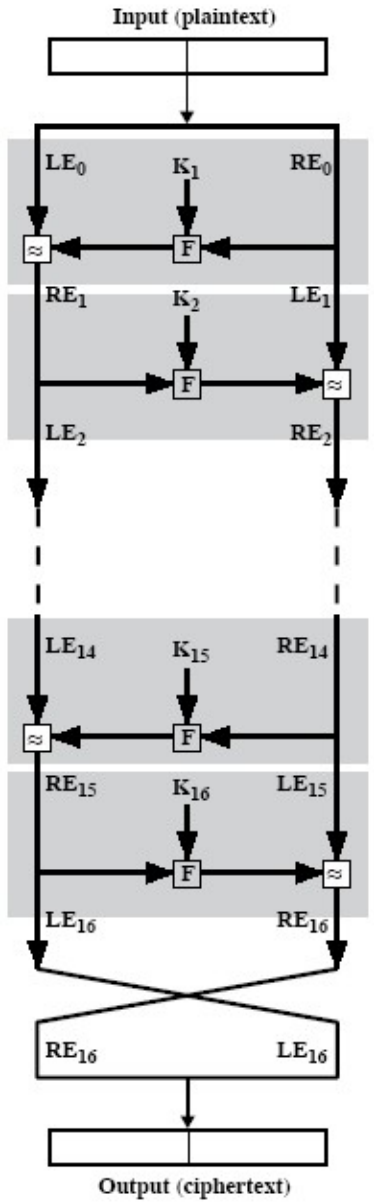
$LD_1 = RD_0 = LE_{16} = RE_{15}$

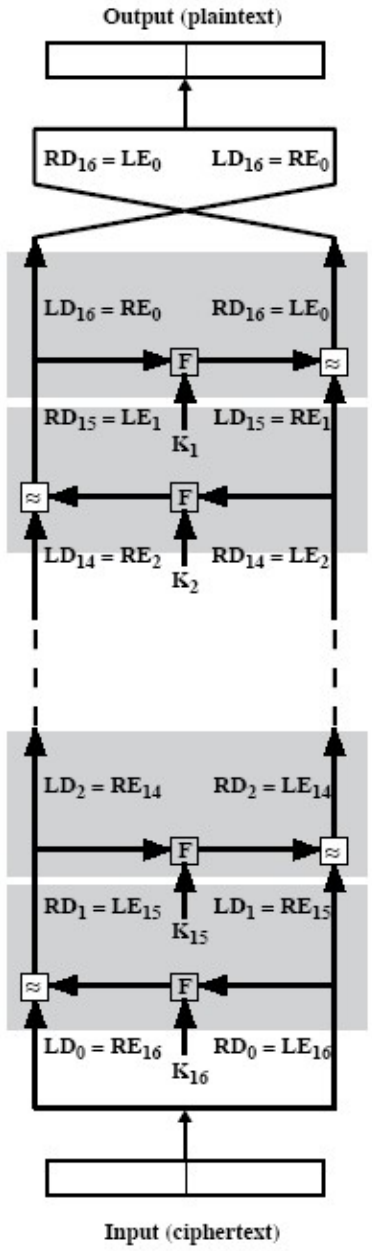
$RD_1 = LD_0 \oplus \mathrm{F}(RD_0, K_{16})$

$= RE_{16} \oplus \mathrm{F}(RE_{15}, K_{16})$

$= LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16}) \oplus \mathrm{F}(RE_{15}, K_{16})$

$= LE_{15}$





Feistel Decryption Algorithm

To show $LD_i \,||\, RD_i = RE_{16-i} \,||\, LE_{16-i}$

Proof:

$LE_{16} = RE_{15}$

$RE_{16} = LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16})$

$LD_1 = RD_0 = LE_{16} = RE_{15}$

$RD_1 = LD_0 \oplus \mathrm{F}(RD_0, K_{16})$

$= RE_{16} \oplus \mathrm{F}(RE_{15}, K_{16})$

$= LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16}) \oplus \mathrm{F}(RE_{15}, K_{16})$

$= LE_{15}$

$LE_i = RE_{i-1}$

$RE_i = LE_{i-1} \oplus \mathrm{F}(RE_{i-1}, K_i)$

⇩

$RE_{i-1} = LE_i$

$LE_{i-1} = RE_i \oplus \mathrm{F}(RE_{i-1}, K_i)$

$= RE_i \oplus \mathrm{F}(LE_i, K_i)$

Let $C = RE_{16} \,||\, LE_{16} = LD_0 \,||\, RD_0$

Then $RD_{16} \,||\, LD_{16} = LE_0 \,||\, RE_0 = P$

DES History

- In 1973, NBS (NIST) issues a public request for proposals for a national cipher standard, which must be
  - Secure
  - Public
  - Completely specified
  - Easy to understand
  - Available to all users
  - Economic and efficient in hardware
  - Able to be validated
  - Exportable
- IBM submitted LUCIFER (Feistel) (which was redesigned to become the DES)
- In 1977, adopted by NBS (NIST) as DES (Data Encryption Standard, Federal Information Processing Standard 46 (FIPS PUB 46))

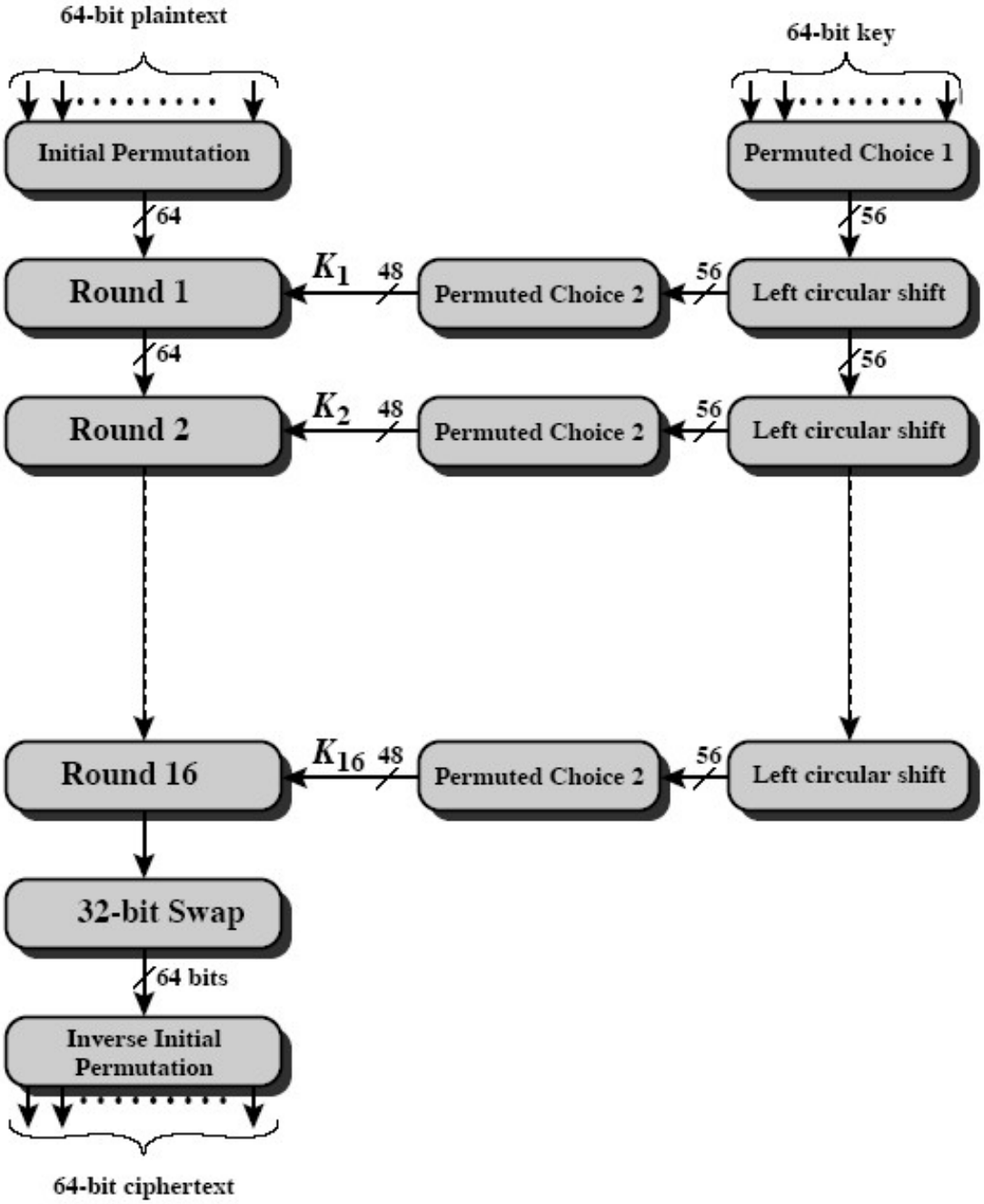
DES History

- Chronology
  http://en.wikipedia.org/wiki/Data_Encryption_Standard
  - 1973: NBS publishes a first request for a standard encryption algorithm
  - 1974: NBS publishes a second request for encryption algorithms
  - 1975: DES is published in the Federal Register for comment
  - 1976: First and second workshop on DES
  - 1976: DES is approved as a standard
  - 1977: DES is published as a FIPS standard FIPS PUB 46
  - 1983: DES reaffirmed for the first time
  - 1986: Videocipher II, a TV satellite scrambling system based upon DES begins use by HBO
  - 1988: DES is reaffirmed for the second time as FIPS 46-1, superseding FIPS PUB 46
  - 1992: Biham and Shamir publish the first theoretical attack with less complexity than brute force: differential cryptanalysis. However, it requires an unrealistic 2^{47} chosen plaintexts
  - 1993: DES is reaffirmed for the third time as FIPS 46-2

DES History

- 1994: The first experimental cryptanalysis of DES is performed using linear cryptanalysis (Matsui, 1994)
- 1997: The DESCHALL Project breaks a message encrypted with DES for the first time in public
- 1998: The Electronic Frontier Foundation (EFF)'s DES cracker (Deep Crack) breaks a DES key in 56 hours
- 1999: Together, Deep Crack and distributed.net break a DES key in 22 hours and 15 minutes
- 1999: DES is reaffirmed for for the fourth time as FIPS 46-3, which specifies the preferred use of Triple DES, with single DES permitted only in legacy systems
- 2001: The Advanced Encryption Standard is published in FIPS 197
- 2002: The AES standard becomes effective
- 2004: The withdrawal of FIPS 46-3 (and a couple of related standards) is proposed in the Federal Register
- 2005: NIST withdraws FIPS 46-3

Overall Scheme of DES Encryption



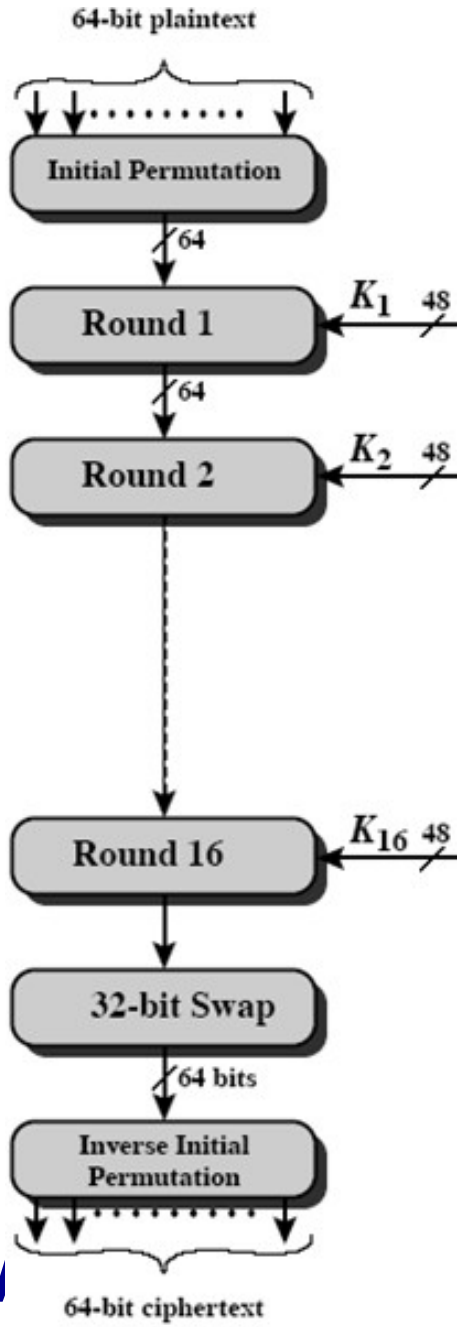
Initial / Inverse Initial Permutation

(a) Initial Permutation (IP)

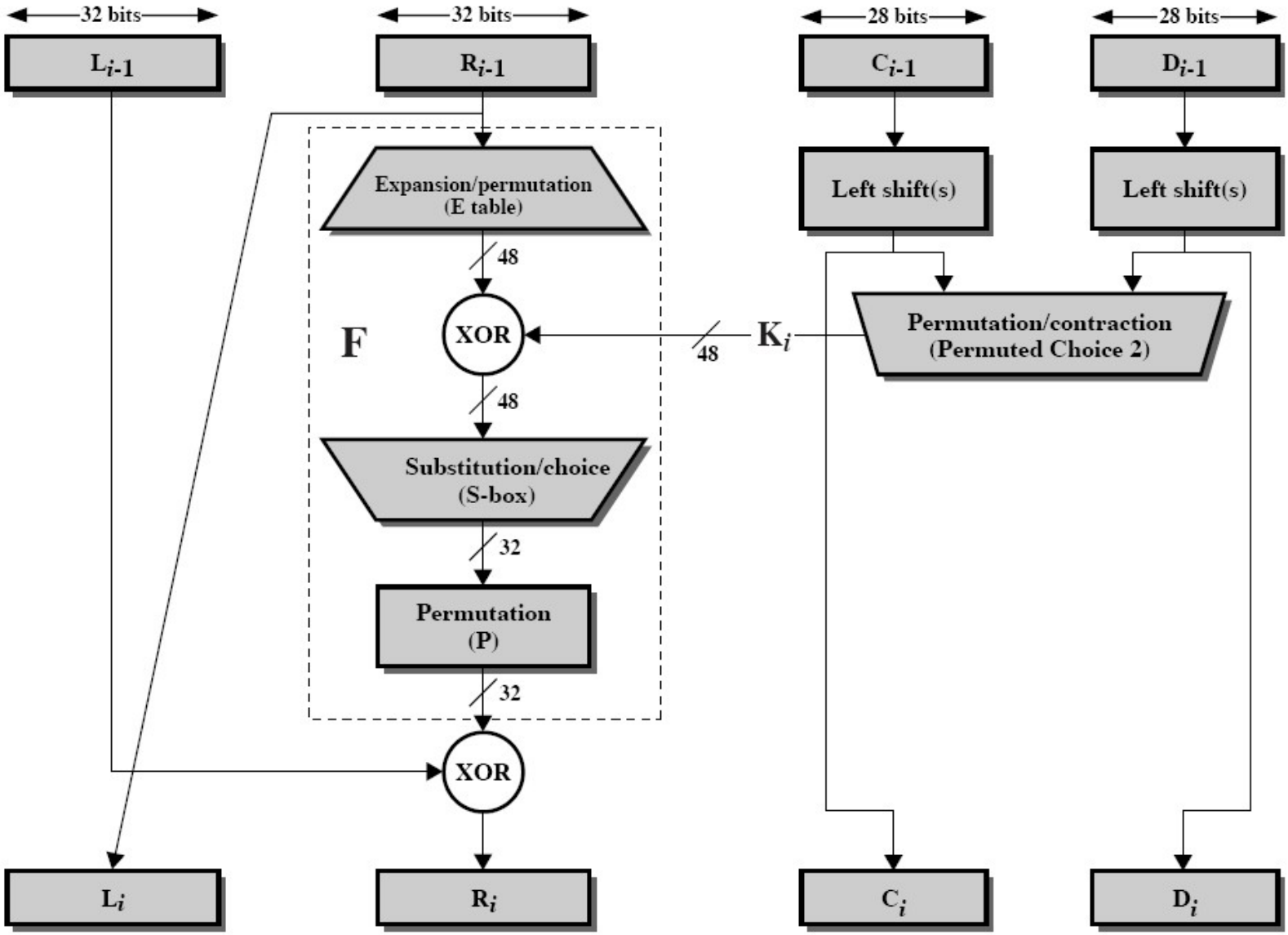
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(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



Single Round of DES

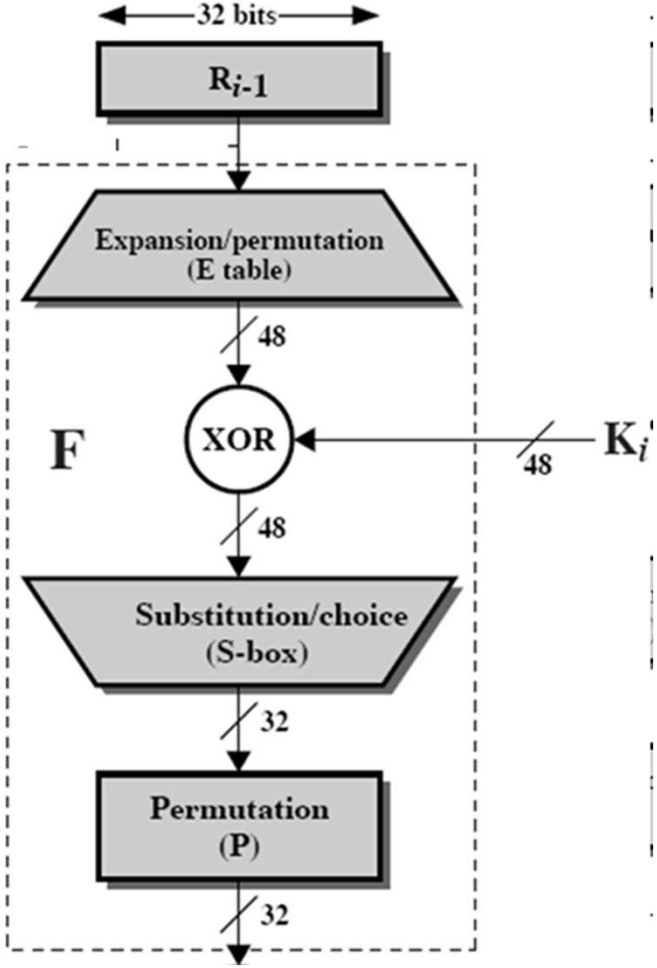


COEP TECHNOLOGICAL UNIVERSITY

Shivajinagar, Pune-411 005

(A Unitary Technological University of Govt. of Maharashtra)

DES Round Function



(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

F(R,K) and S-Boxes

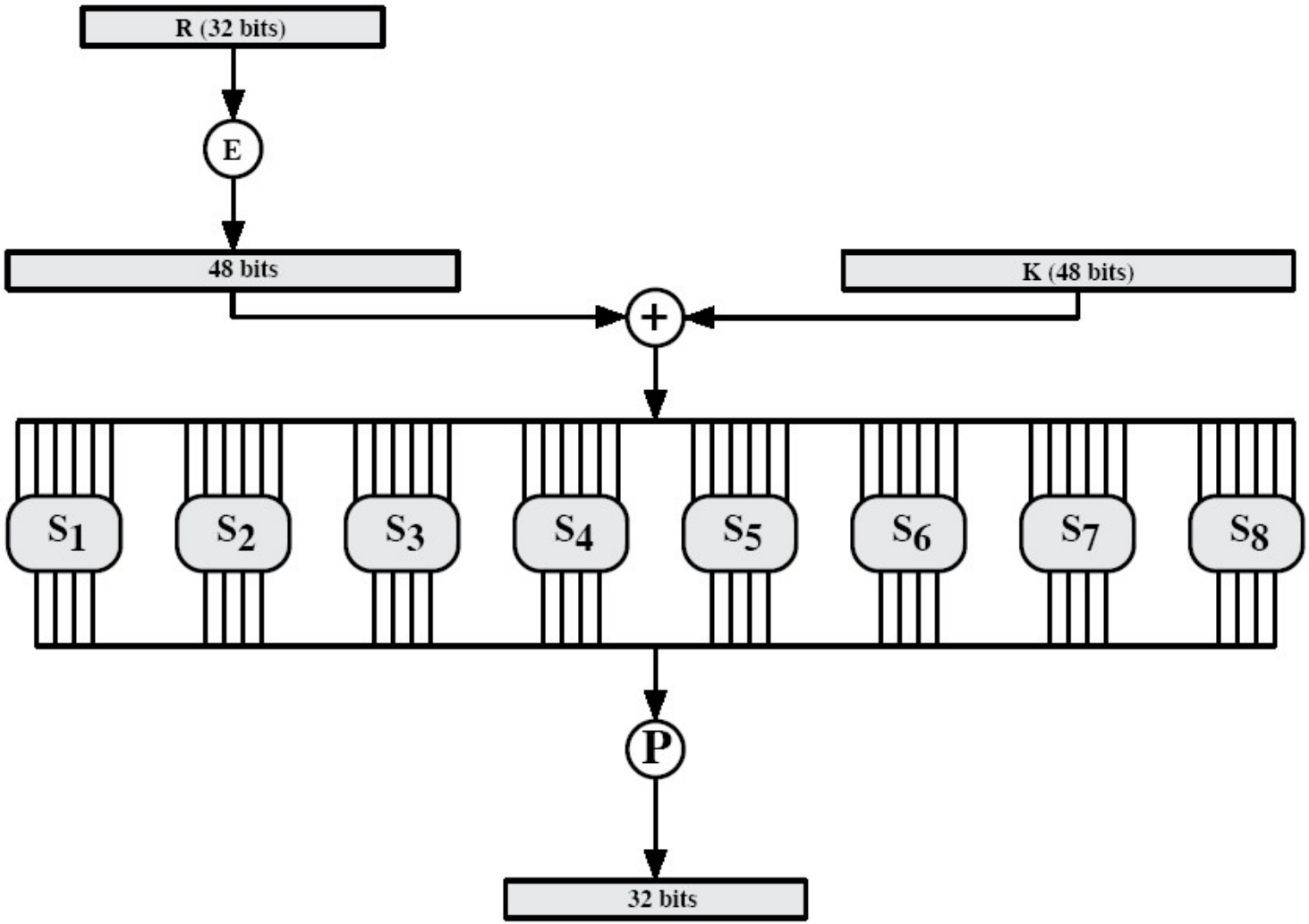


Figure 3.6 Calculation of F(R, K)

Definition of DES S-Boxes

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

- S-box in DES is a bijective function that maps each input to a unique output
- No two inputs produce the same output, and some input reaches every possible output
- This ensures that the S-box can be inverted for decryption

S-Box Details

- Eight S-boxes, each maps 6 bits to 4 bits
- One S-box contains 64 entries, each with 4 bits
- Can be viewed as four permutations of {0, …, 15}
- Final permutation is applied on the combined bits from the 8 tables

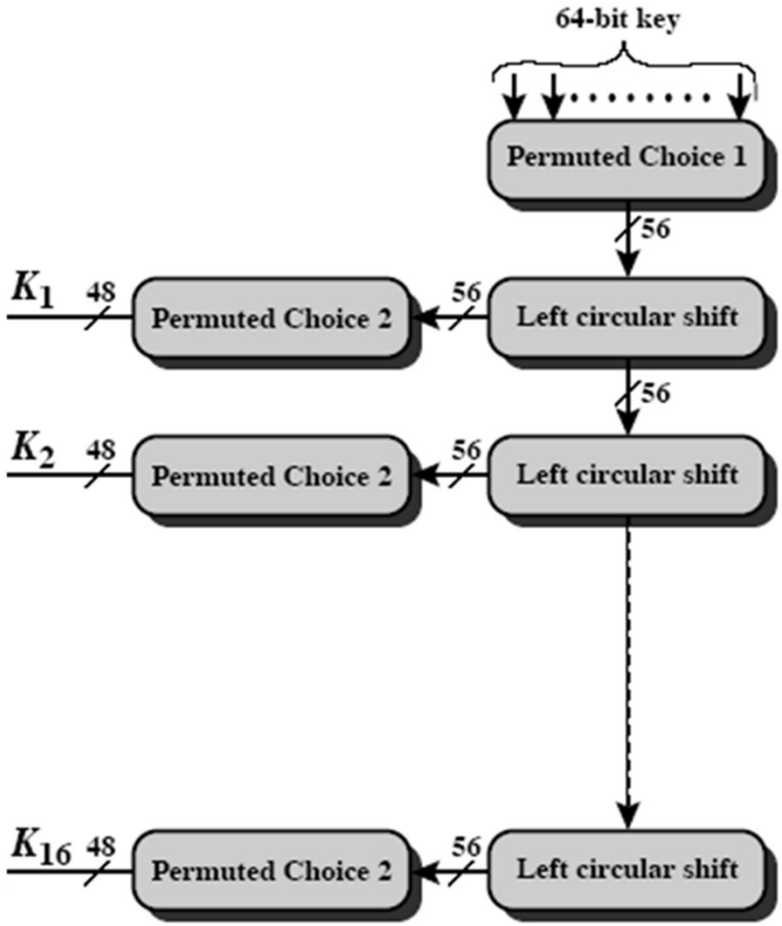
Column

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

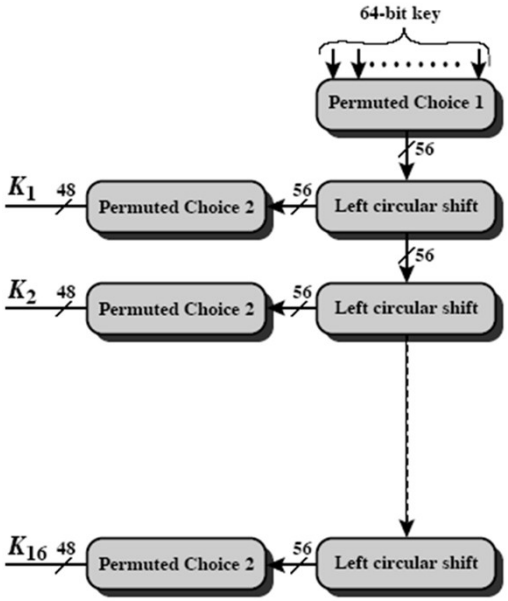
Row

DES Key Generation

- Parity bits (8, 16, ..., 64) are discarded (out of 64-bit key)
- 56-bit key is split into 28-bit L and R
- 16 48-bit subkeys (K_1, K_2, ... , K_{16}) are generated by various circular left shifts of L and R
- Bits are permuted and selected



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)

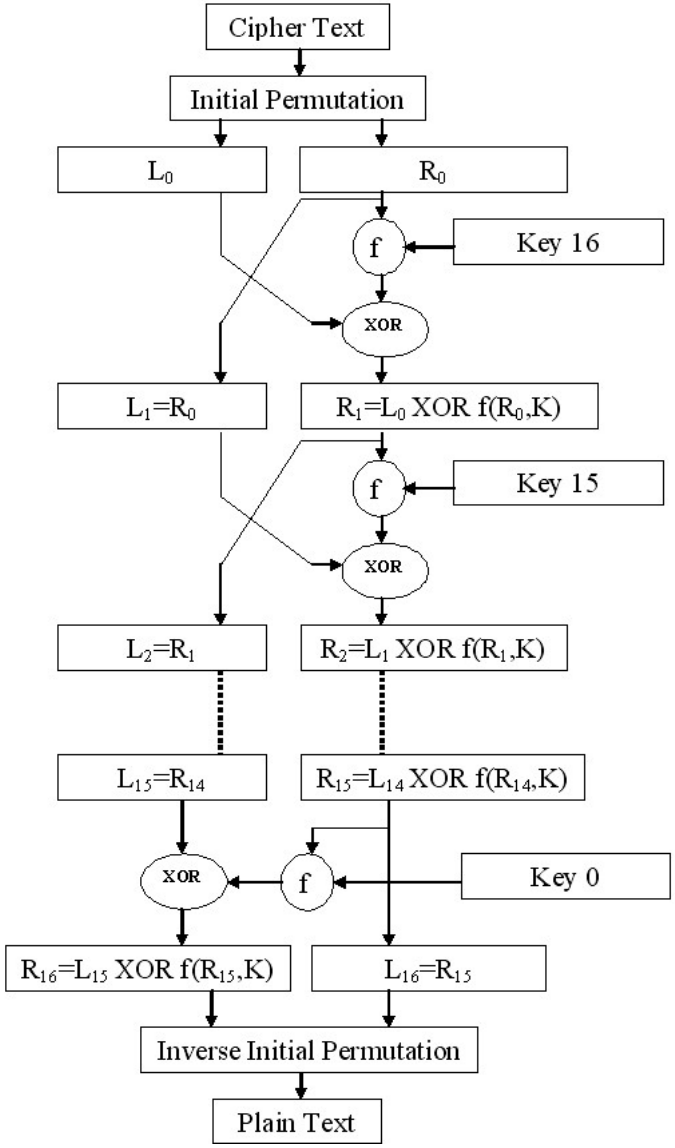
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(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

DES Decryption

- Just as in Feistel cipher, apply the subkeys in reverse order



DES Avalanche Effect

- In any good cipher, any change in the key or plaintext, no matter how large or small, should change approximately half the ciphertext bits
- Examples
  - (a) Change one bit in the plaintext with the same key
  - (b) Change one bit in the key with the same plaintext
  - After 3 or 4 rounds, approximately half of the ciphertext bits are changed
  - After 16 rounds, a lot of scrambling has taken place

Table 3.5 Avalanche Effect in DES

(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

Strength of DES

- 56-bit key is susceptible to exhaustive key search due to rapid advances in computing speed
- Have demonstrated key breaking
  - 1997 on a large network of computers in a few months
  - 1998 on dedicated H/W in a few days (www.eff.org/descracker)
    - EFF (Electronic Frontier Foundation) DES Cracker
    - 1536 chips and search 88 billion keys/second
    - $250,000 cost, won the RSA DES Challenge II Contest in less than 3 days (56 hours)
  - 1999 above combined in 22 hours !! (DES Cracker + 100,000 computers)
  - DES also theoretically broken using Differential or Linear Cryptanalysis

Attacks on DES

- Standard attacks
  - exhaustive key search
  - dictionary attack
  - differential cryptanalysis
  - linear cryptanalysis

- Side channel attacks against implementations.
  - Timing attacks
  - Power consumption attacks
  - Fault injection attacks

DES Weak Keys

- DES has 4 weak keys (64-bit)
  - 01010101 01010101
  - FEFEFEFE FEFEFEFE
  - E0E0E0E0 F1F1F1F1
  - 1F1F1F1F 0E0E0E0E
- Using weak keys, the outcome of the Permuted Choice 1 (PC1) in the DES key schedule leads to round keys being either all zeros, all ones or alternating zero-one patterns
- Since all the subkeys are identical, and DES is a Feistel network, the encryption function becomes self-inverting; that is, encrypting twice with a weak key K produces the original plaintext.– $E_K(E_K(x))=x$ for all x, i.e., the encryption and the decryption are the same
- Weak keys should be avoided at key generation.

Chosen-Plaintext Attacks on Block Ciphers

- Given a Plaintext/Ciphertext pair (P,C), look up C in a table constructed with the following entries
  - (K, E_K[P]) for all possible key K
  - Sort based on the second field (ciphertext)
  - How much time does this take?
- To attack a new key K (under chosen message attacks)
  - Choose P, obtain the ciphertext C, looks up in the table, and finds the corresponding key
  - How much time does this step take?
- Trade off: space for time

Differential Cryptanalysis

- Main idea:
  - This is a chosen plaintext attack,
  - The attacker knows many (plaintext, ciphertext) pairs
  - Difference $\Delta_P = P_1 \oplus P_2$, $\Delta_C = C_1 \oplus C_2$
  - Distribution of Δ_C's given Δ_P may reveal information about the key (certain key bits)
  - After finding several bits, use brute-force for the rest of the bits to find the key

Differential Cryptanalysis of DES

- Surprisingly … DES was resistant to differential cryptanalysis.
- At the time DES was designed, the authors knew about differential cryptanalysis. S-boxes were designed to resist differential cryptanalysis.
- Against 8-round DES, attack requires 2^{38} known plaintext-ciphertext pairs.
- Against 16-round DES, attack requires 2^{47} chosen plaintexts.
- Differential cryptanalysis not effective against DES in practice.

Linear Cryptanalysis of DES

- Introduced in 1993 by M. Matsui
- Instead of looking for isolated points at which a block cipher behaves like something simpler, it involves trying to create a simpler approximation to the block cipher as a whole.
- **Linear Approximation**: The attack relies on finding linear equations that approximate the behavior of the cipher's S-boxes. These equations express a probabilistic relationship between input and output bits.

Linear Cryptanalysis of DES

- Suppose that

$$(*)\ \ \mathbf{Pr}\,[M_{i1} \oplus M_{i2} \oplus \ldots \oplus M_{iu} \oplus C_{j1} \oplus C_{j2} \oplus \ldots \oplus C_{jv} \oplus K_{p1} \oplus k_{p2} \oplus \ldots \oplus k_{pw} = 1\,] = 0.5 + \varepsilon$$

- **Bias in Probability**: For a truly random cipher, the probability of a linear approximation holding true is 50%. However, in DES, certain approximations deviate slightly from this, creating a bias that can be exploited.
- **Key Recovery**: Using the biased approximations, the attacker guesses portions of the key and verifies them against the observed data to recover the full key.

Linear Cryptanalysis of DES

- Then one can recover some key bits given a large number of Plain Text/Cipher Text pairs
- For DES, exists (*) with $\varepsilon=2^{-21}$
- M. Matsui showed (1993/1994) that DES can be broken:
  - 8 rounds: 2^{21} known plaintext
  - 16 rounds: 2^{43} known plaintext, 40 days to generate the pairs (plaintext, ciphertext) and 10 days to find the key
- The attack has no practical implication, requires too many PT/CT pairs.
- Attack is faster than brute force, but it still involves significant computational effort, especially for full 16-round DES.

DES Strength Against Various Attacks

Attack Method	Known	Chosen	Storage complexity	Processing complexity
Exhaustive precomputation	-	1	2^{56}	1
Exhaustive search	1	-	negligible	2^{55}
Linear cryptanalysis	2^{43}	-	For texts	2^{43}
Differential cryptanalysis	-	2^{47}	For texts	2^{47}

The weakest point of DES remains the size of the key (56 bits)!

DES Design Criteria

- Although the standard for DES is public, the design criteria used are classified
- A few have since been made public (released)
- A few have been derived or deduced (by reverse engineering)
- What we know is based mostly on D. Coppersmith, "The Data Encryption Standard (DES) and Its Strength Against Attacks," *IBM J. of R. and D.* (May 1994)
  - 7 Criteria for S-boxes provide for
    - Non-linearity
    - Resistance to differential cryptanalysis
    - Good confusion
  - 3 Criteria for permutation P provide for
    - Increased diffusion

DES S-Box Design Criteria

- S-box is the only source of nonlinearity in DES
- No S-box output bit should be too close to a linear function of the input bits (or any subset of them)
  - if we select any output bit and any subset of the input bits, then the fraction of inputs for which the output bit is the xor of the input bits should be close to ½
- Each row of an S-box should be a permutation
- If two inputs to an S-box differ in exactly one bit, then the outputs must differ in at least two bits
- If two inputs to an S-box differ in exactly the middle two bits, then the outputs must differ in at least two bits
- If two inputs to an S-box differ in their first two bits and are identical in their last two bits, then the two outputs should not be the same
- The first criteria is for *nonlinearity*, and the others have mostly to do with providing good *confusion* properties and resistance to *differential cryptanalysis*

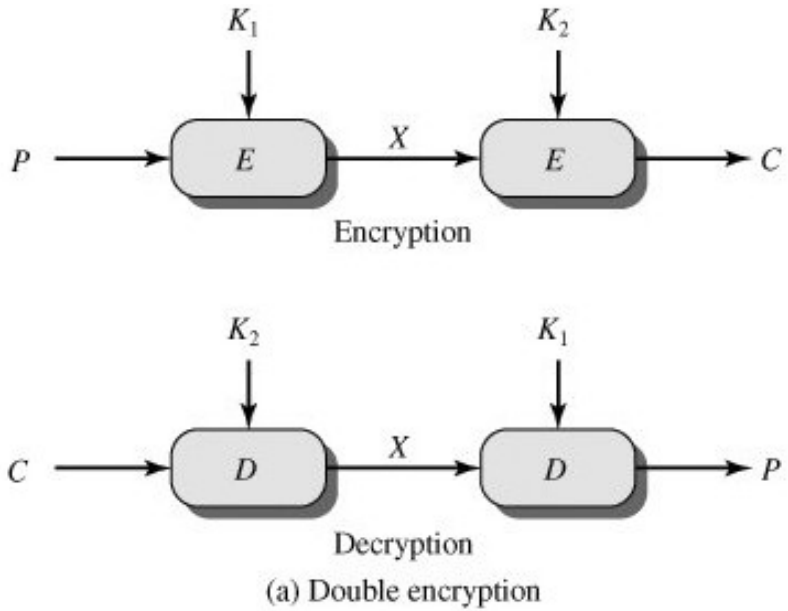
Some Other Design Criteria

- Some design criteria for P to increase the *diffusion*
- *16 rounds* is required to thwart the differential cryptanalysis attack (with 16 rounds differential cryptanalysis attack is less efficient than the brute-force key search attack)
- Additional design criteria for F (and hence for S-box)
  - Strict Avalanche Criterion (SAC): Any output bit j should change with probability ½ when any input bit i is changed for all i, j
  - Bit Independence Criterion (BIC): Output bits j and k should change independently when any single input bit i is changed, for all i, j, k
- Key schedule should guarantee key/ciphertext SAC and BIC
- DES Avalanche Property
  - Changing a single bit in the input changes on average half of the bits at the output
- DES Completeness Property
  - Every output bit should be a complex function of all input bits (and not just a subset of input bits)

Multiple Encryption with DES

- DES is vulnerable to brute force attack
- Alternative block cipher that makes use of DES software/equipment/knowledge: encrypt multiple times with different keys
- Options:
  - 1. Double DES: not much better than single DES
  - 2. Triple DES (3DES) with 2 keys: brute force 2^{112}
  - 3. Triple DES with 3 keys: brute force 2^{168}

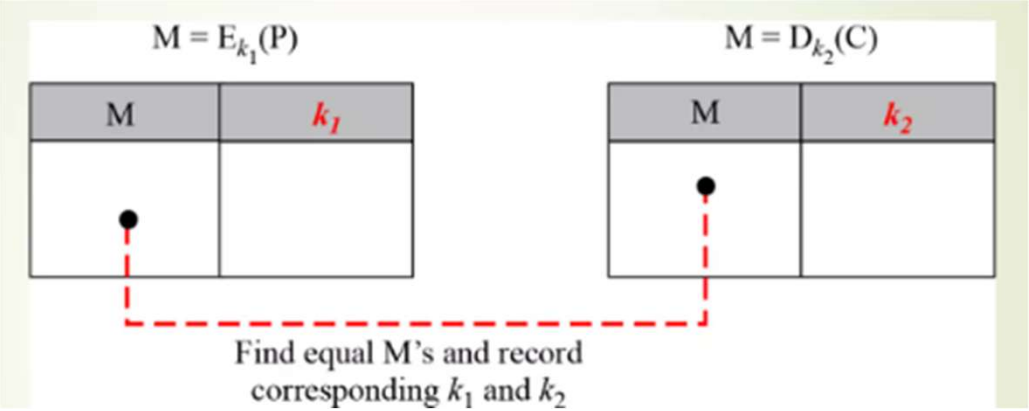
Double-DES



(a) Double encryption

- Could use 2 DES encrypts on each block
  - $C = E_{K2}E_{K1}(P))$
- Issue of reduction to single stage
  - There might be a single key that is equivalent to using 2 keys as above, not likely, but only finally proved in 1992

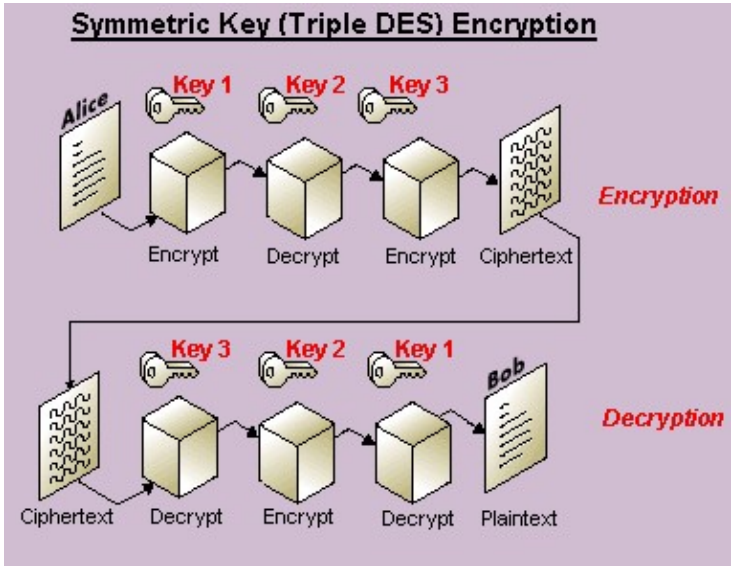
Double-DES: “meet-in-the-middle” Attack



- since $X = E_{K1}(P) = D_{K2}(C)$
- attack by encrypting P with all keys and store
- then decrypt C with keys and match X value
- can show only to take $O(2^{56})$ steps to find the key

Triple Encryption

- 2 keys, 112 bits
- 3 keys, 168 bits



- Why E-D-E?
  - With EEE the inverse permutation at the end of the first encryption would be canceled out by the initial permutation of the second encryption (same for inverse of second and initial of third encryption)
  - EEE results in slightly reduced brute-force attack time

Triple-DES with Two-Keys

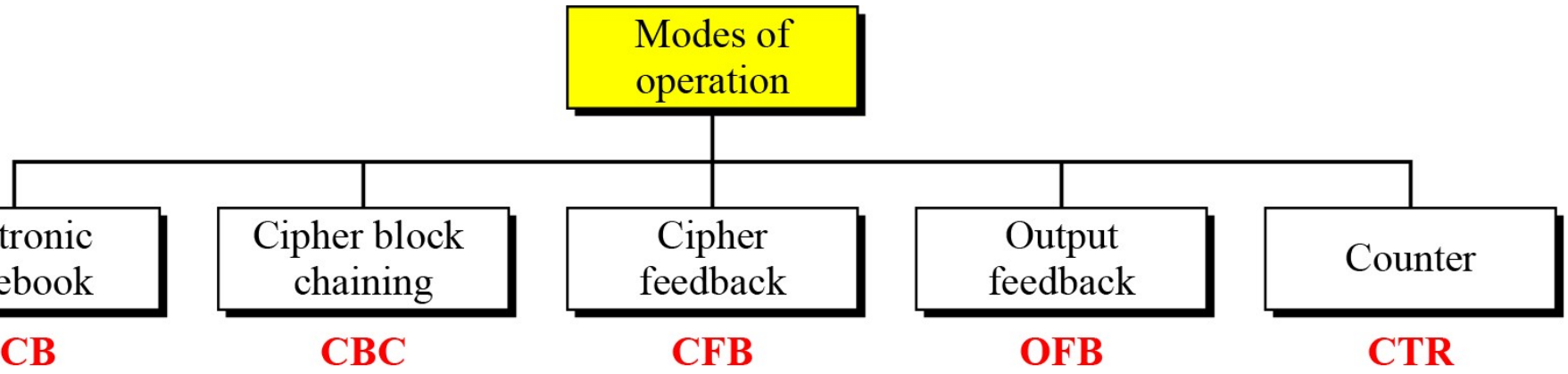
- Must use 3 operations (encryptions)
  - would seem to need 3 distinct keys
- but can use 2 keys if E-D-E sequence is used
  - $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - Encrypt and decrypt have equivalent in security strength
  - if K1=K2 then can work with single DES
- Standardized in ANSI X9.17 and ISO8732
- No current known practical attacks
  - Cost of a brute-force key search $O(2^{112})$ $(=5*10^{33})$
  - Cost of differential cryptanalysis compared to single DES, exceeding 10^{52}

Triple-DES with Three-Keys

- Although are no practical attacks on two-key Triple-DES have some indications
- Can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}(D_{K2}(E_{K1}(P)))$
- Has been adopted by some Internet applications, eg PGP, S/MIME

Modes of Operation

- Block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks with 56-bit key
- Need some way to en/decrypt arbitrary amounts of data (may be large) in practise
- **ANSI X3.106-1983 Modes of Use** (now FIPS 81) defines 4 possible modes
- Subsequently 5 modes defined for AES and DES
- Have **block** and **stream** modes

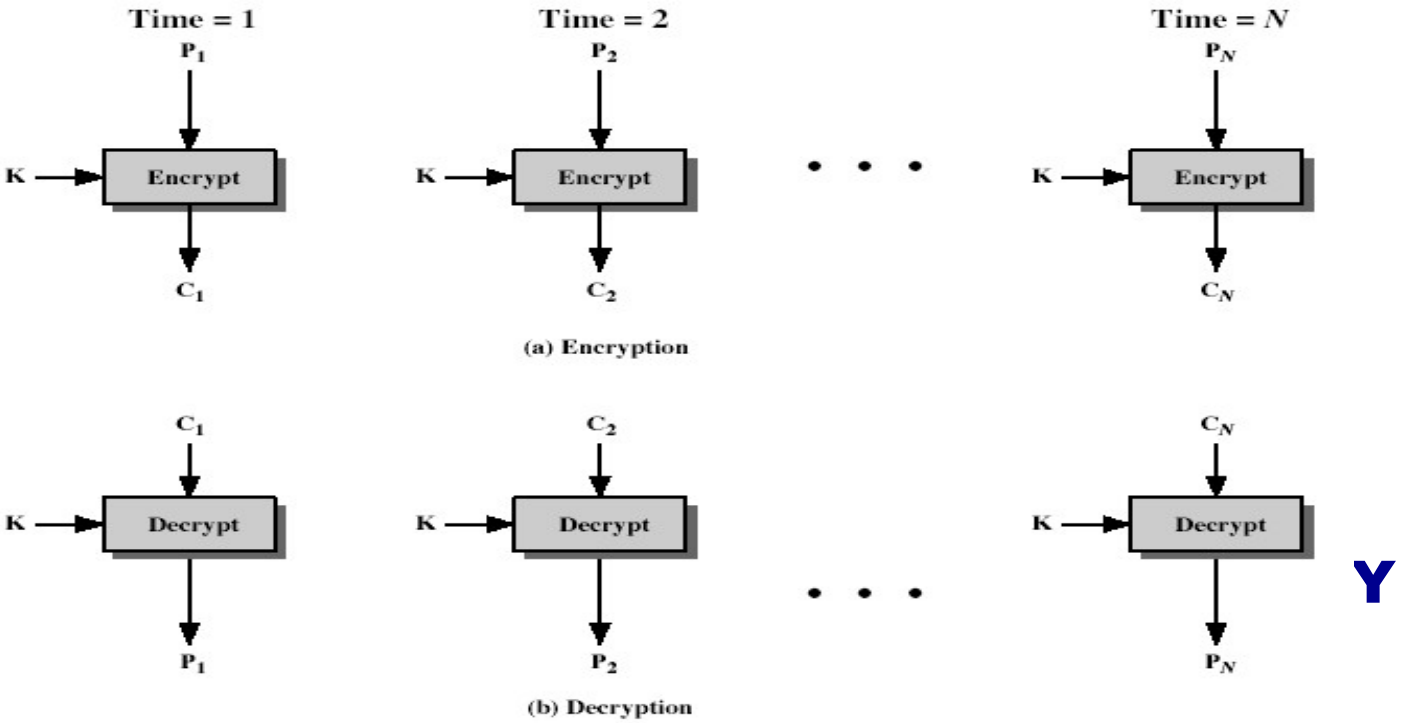


Electronic Codebook Book (ECB)

- Message is broken into independent blocks (like a codebook, hence name) which are encrypted
- Each block is encoded independently of the other blocks

$$C_i = DES_{K1}(P_i)$$

- Uses: secure transmission of single values



Advantages and Limitations of ECB

- Deterministic: the same data block gets encrypted the same way; this reveals patterns of data when a data block repeats
- Malleable: reordering ciphertext results in reordered plaintext
- Weakness is due to the encrypted message blocks being independent
- ECB is not appropriate for large quantities of data
  - since repetitions can be seen, esp. with graphics
  - because the blocks can be shuffled/inserted without affecting the en/decryption of each block
- It's main use is to send one or a very few blocks (eg. session keys encrypted using a master key)

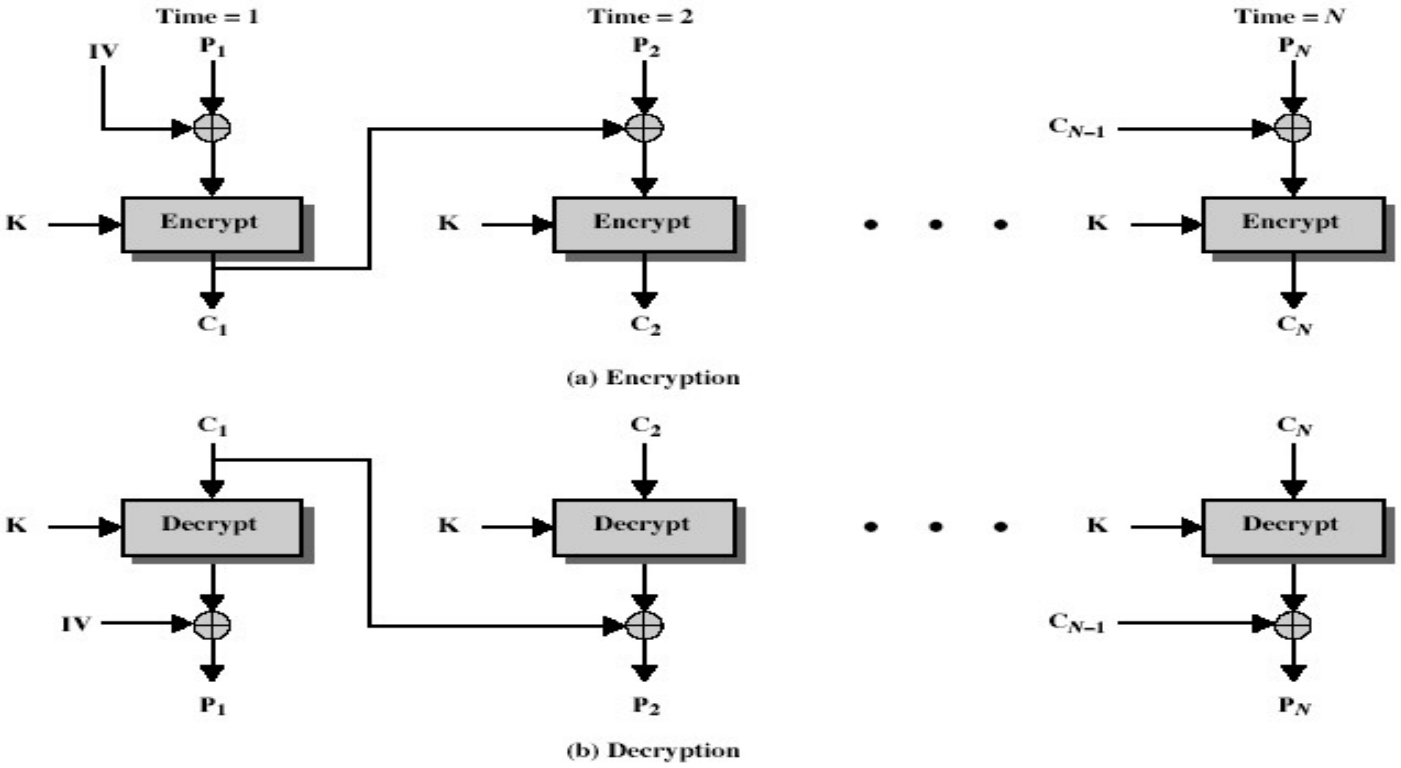
Cipher Block Chaining (CBC)

- Message is broken into blocks. Each previous cipher blocks is chained with current plaintext block
- Use Initial Vector (IV) to start process

$C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$

$C_{-1} = IV$

- Uses: bulk data encryption, authentication



COEP Tech

Message Padding

- At end of message must handle a possible last short block
  - which is not as large as blocksize of cipher
  - pad either with known non-data value (eg nulls)
  - or pad last block along with count of pad size
    - eg. [b1 b2 b3 0 0 0 0 5]
    - means have 3 data bytes, then 5 bytes pad+count
  - this may require an extra entire block over those in message
- there are other, more esoteric modes, which avoid the need for an extra block

Advantages and Limitations of CBC

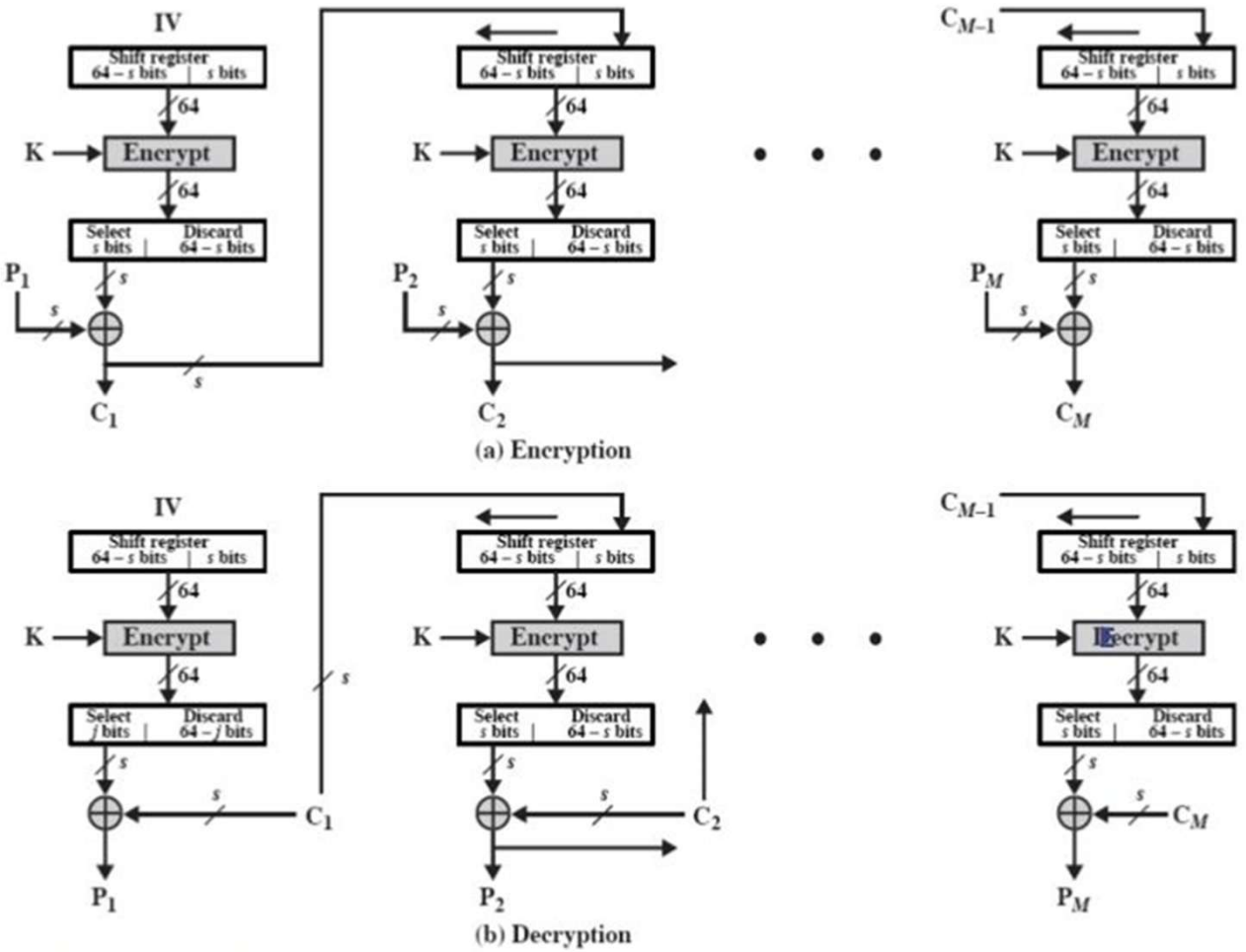
- Randomized encryption: repeated text gets mapped to different encrypted data

$$C_1 = Enc_k(P_1 \oplus IV)$$

$$C_i = Enc_k(P_i \oplus C_{i-1}), \quad 1 < i \leq nb$$

- A ciphertext block depends on **all** blocks before it
- Any change to a block affects all following ciphertext blocks
- Sequential encryption, cannot use parallel hardware
- Need **Initialization Vector** (IV) known to sender and receiver
  - IV must either be a fixed or must be sent encrypted in ECB mode before rest of message

Cipher FeedBack (CFB)



Note that the block cipher is used in **encryption** mode at **both** ends

Cipher FeedBack (CFB)

- Message is treated as a stream of bits
- Added to the output of the block cipher
- Result is feed back for next stage
- Standard allows any number of bit (1,8, 64 or 128 etc) to be feed back
  - denoted CFB-1, CFB-8, CFB-64, CFB-128 etc
- Most efficient to use all bits in block (64 or 128)

$$C_i = P_i \text{ XOR } DES_{K1}(C_{i-1})$$

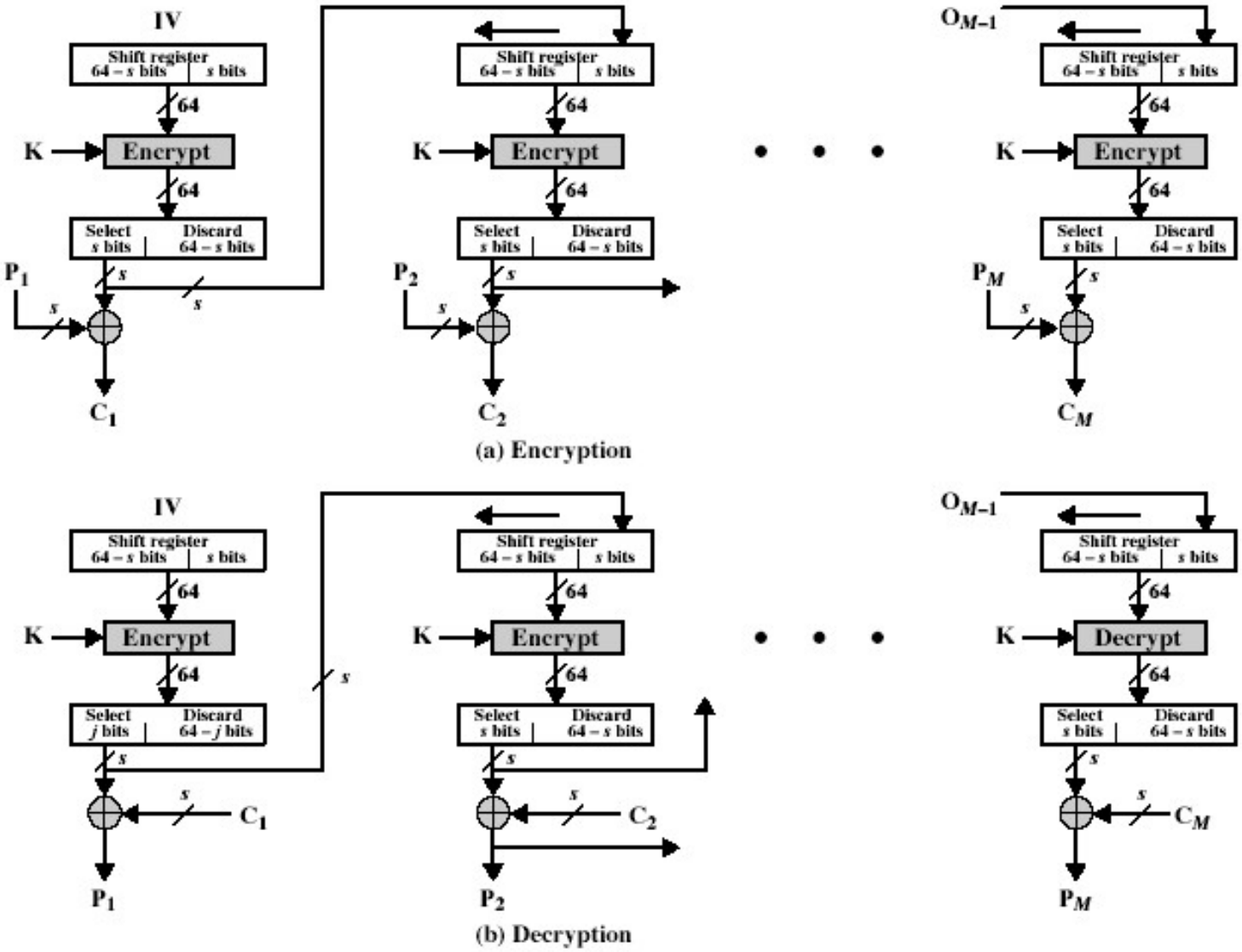
$$C_{-1} = IV$$

- uses: stream data encryption, authentication

Advantages and Limitations of CFB

- Appropriate when data arrives in bits/bytes (stream mode)
- Randomized encryption
- A ciphertext block depends on all preceding plaintext blocks; reorder affects decryption
- Errors propagate for several blocks after the error, but the mode is self-synchronizing (like CBC).
- Decreased throughput.– Can vary the number of bits feed back, trading off throughput for ease of use
- If it is used over a "noisy" link, errors propagate for several blocks after the error

Output FeedBack (OFB)



Output FeedBack (OFB)

- Message is treated as a stream of bits
- Output of cipher is added to the message
- Output is then feedback
- Feedback is independent of the message
- Can be computed in advance

$$C_i = P_i \text{ XOR } O_i$$

$$O_i = DES_{K1}(O_{i-1})$$

$$O_{-1} = IV$$

- Good for bursty traffic
- Uses: Stream encryption on noisy channels
  - any bit error only affects a single bit

Advantages and Limitations of OFB

- More vulnerable to message stream modification
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
  - (otherwise the 2 ciphertexts can be combined, cancelling these bits, and leaving a "book" cipher to solve)
- Sender and receiver must remain in sync
- Originally specified with m-bit feedback
- Subsequent research has shown that only **full block feedback** (ie CFB-64 or CFB-128) should ever be used

Counter (CTR)

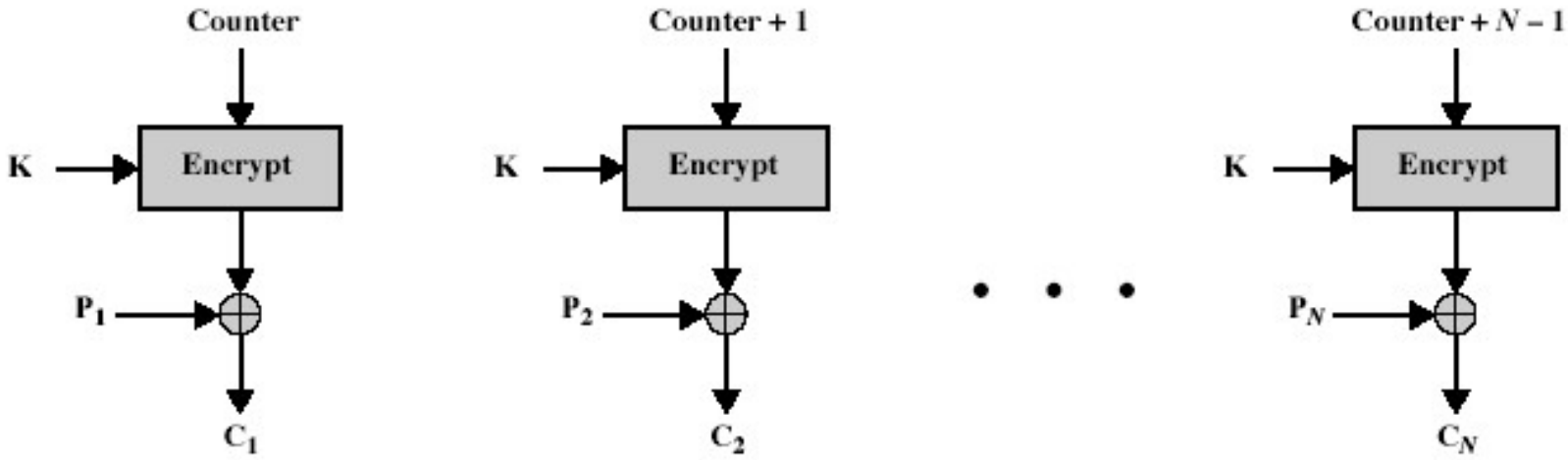
- A “new” mode, though proposed early on
- Similar to OFB but encrypts counter value rather than any feedback value
- Must have a different key and counter value for every plaintext block (never reused)

$$C_i = P_i \text{ XOR } O_i$$

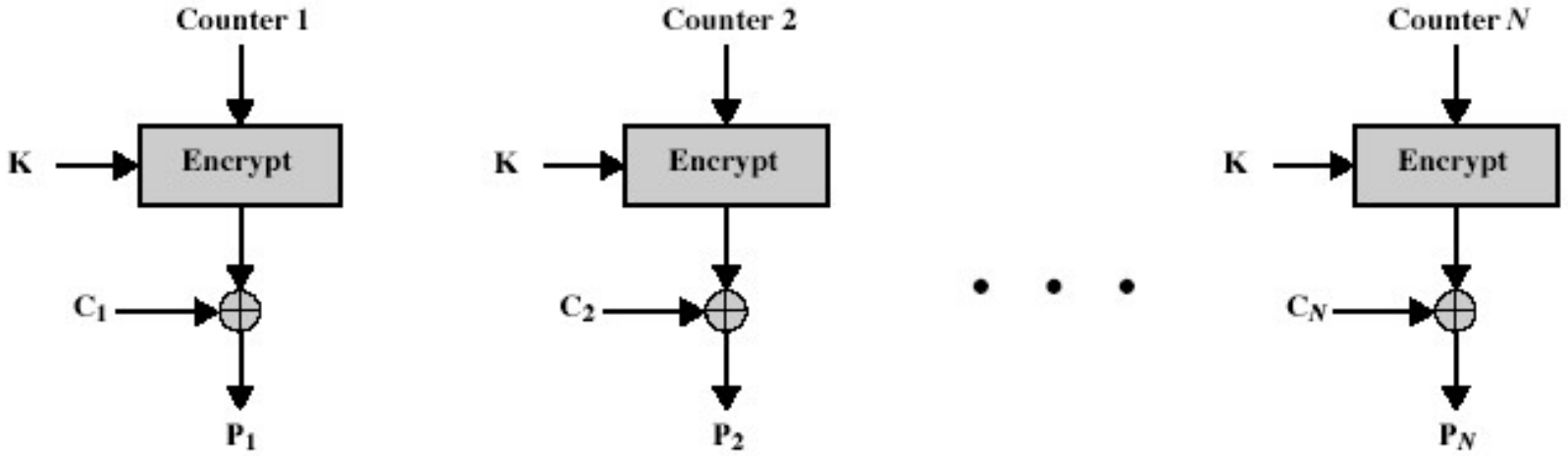
$$O_i = DES_{K1}(i)$$

- Uses: high-speed network encryptions

Counter (CTR)



(a) Encryption



(b) Decryption

Advantages and Limitations of CTR

- Efficiency
  - can do parallel encryptions in hardware or software
  - Preprocessing: the encryption part can be done offline and when the message is known, just do the XOR.
  - Good for bursty high speed links
- Random access: decryption of a block can be done in random order, very useful for hard-disk encryption.
- Provable security (good as other modes)
- But must ensure never reuse key/counter values, otherwise could break (cf OFB)

Error Propagation in Cyber Modes

- A bit error is the substitution of a ‘0’ bit for a ‘1’ bit, or vice versa
- These errors originate in the transmission channel as a consequence of interference and noise
- Errors in the cryptogram produce errors in the decrypted plaintext. This phenomenon is called error propagation.

Error Propagation Summary

Mode of Operation	Error Propagation	Effect of Bit Errors in C_j
ECB	Errors in one ciphertext block do not propagate	RBE in Pj
CBC	Errors in one block propagate to two blocks– one bit error in C_j affects all bits in P_j and one bit in P_{j+1}	RBE in Pj SBE in Pj+1
CFB	Errors propagate for several blocks after the error	SBE in P j RBE in P j+1, P j+2, ... P j+w
OFB	Bit errors do not propagate	SBE in Pj
CTR	Bit errors do not propagate	SBE in Pj

The effect of the error bit $c_{i,j}$ in the block $C_j = (c_{1,j}, c_{2,j}, ..., c_{b,j})$
RBE (random bit errors) means that an individual error bit $c_{i,j}$ affects randomly all bits in the plaintext block P_j
SBE (specific bit errors) means that an individual error bit $c_{i,j}$ produces an individual error bit $p_{i,j}$ in the same bit positions