Block Ciphers

- Map n-bit plaintext blocks to n-bit ciphertext blocks (n = block length).
- For n-bit plaintext and ciphertext blocks and a fixed key, the encryption function is a bijection;
- $E : P_n \times K \rightarrow C_n$ s.t. for all key $k \in K$, $E(x, k)$ is an invertible mapping, written $E_k(x)$.
- The inverse mapping is the decryption function, $y = D_k(x)$ denotes the decryption of plaintext x under k.

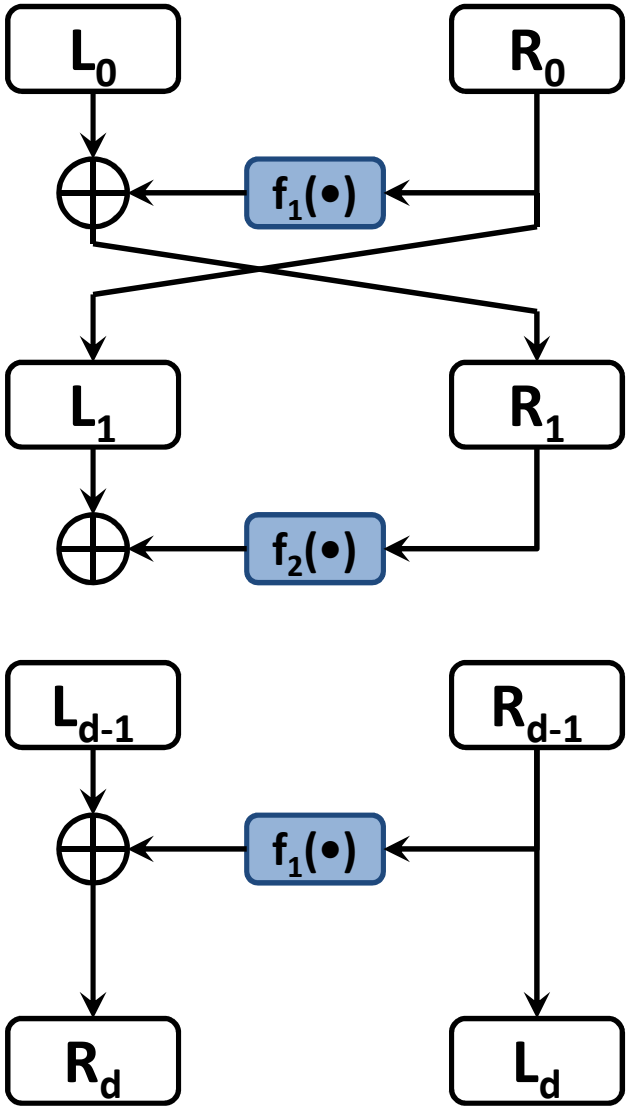
Block Ciphers Features

- Block size: in general *larger* block sizes mean *greater* security.
- Key size: *larger* key size means *greater* security (larger key space).
- Number of rounds: multiple rounds offer increasing security.
- Encryption modes: define how messages larger than the block size are encrypted, *very important* for the security of the encrypted message.

Feistel Network

- Several block ciphers are based on the structure proposed by *Feistel* in 1973
- A *Feistel Network* is fully specified given
  - the *block size*: n = 2w
  - *number of rounds*: d
  - d *round functions* $f_1, \ldots, f_d: \{0,1\}^w \rightarrow \{0,1\}^w$
- Used in DES, IDEA, RC5 (Rivest's Cipher n. 5), and many other block ciphers.
- Not used in AES

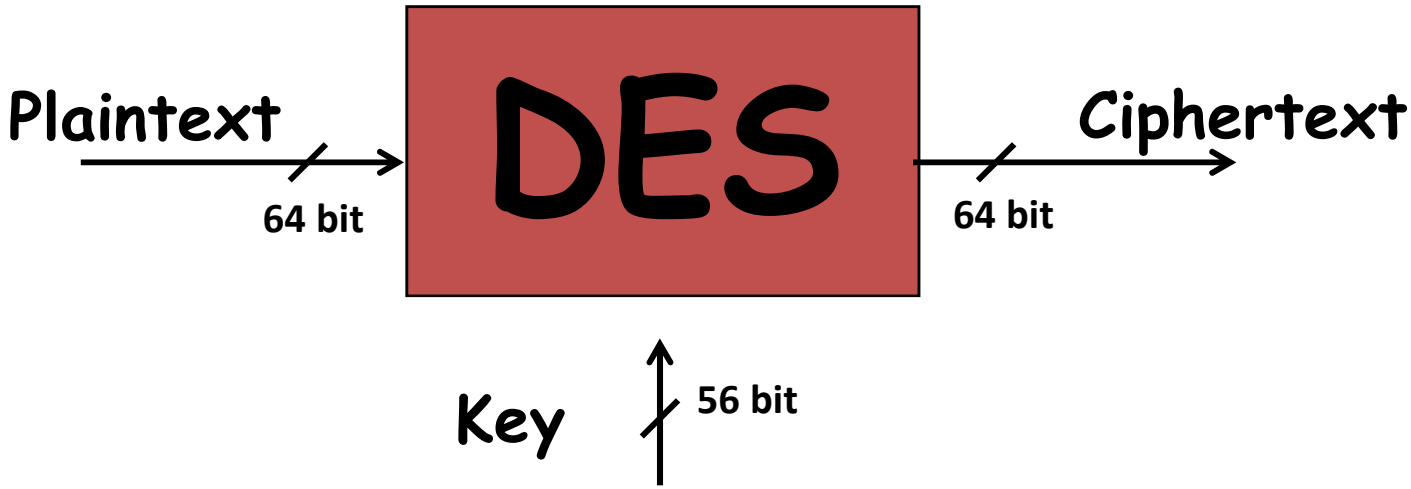
Feistel Network



- **Encryption**:
  - $L_1 = R_0 \quad R_1 = L_0 \oplus f_1(R_0)$
  - $L_2 = R_1 \quad R_2 = L_1 \oplus f_2(R_1)$
  - …
  - $L_d = R_{d-1} \quad R_d = L_{d-1} \oplus f_d(R_{d-1})$
- **Decryption**:
  - $R_{d-1} = L_d \quad L_{d-1} = R_d \oplus f_d(L_d)$
  - …
  - $R_0 = L_1; \quad L_0 = R_1 \oplus f_1(L_1)$

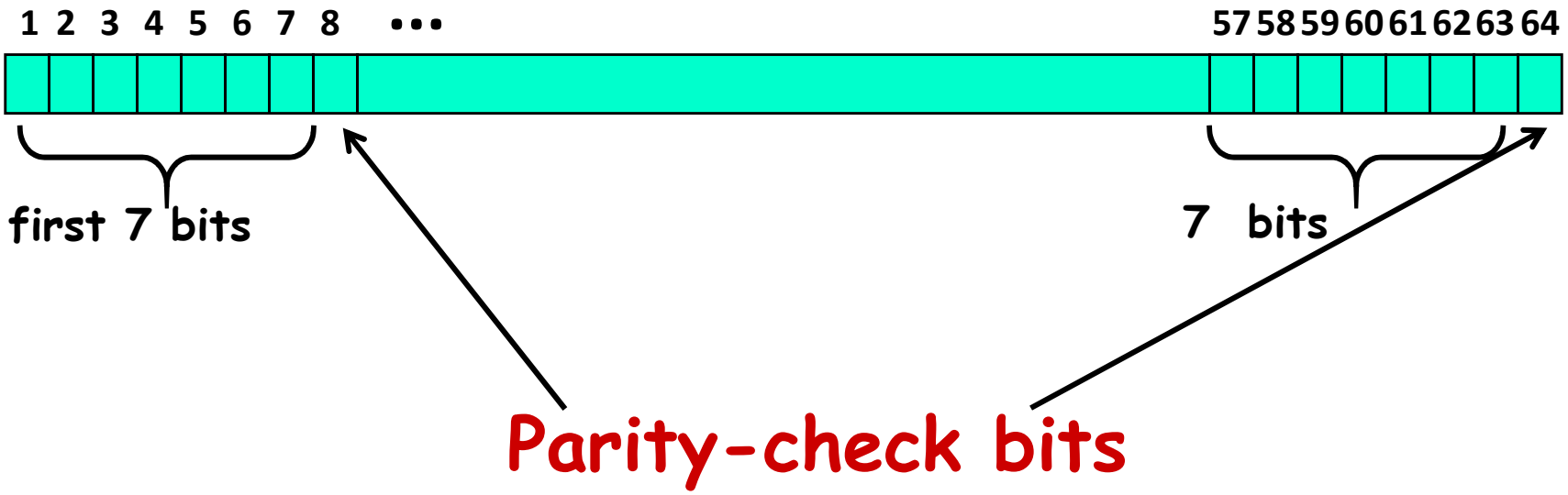
DES Features

- Features:
  - Block size = 64 bits
  - Key size = 56 bits (in reality, 64 bits, but 8 are used as parity-check bits for error control, see next slide)
  - Number of rounds = 16
  - 16 intermediary keys, each 48 bits



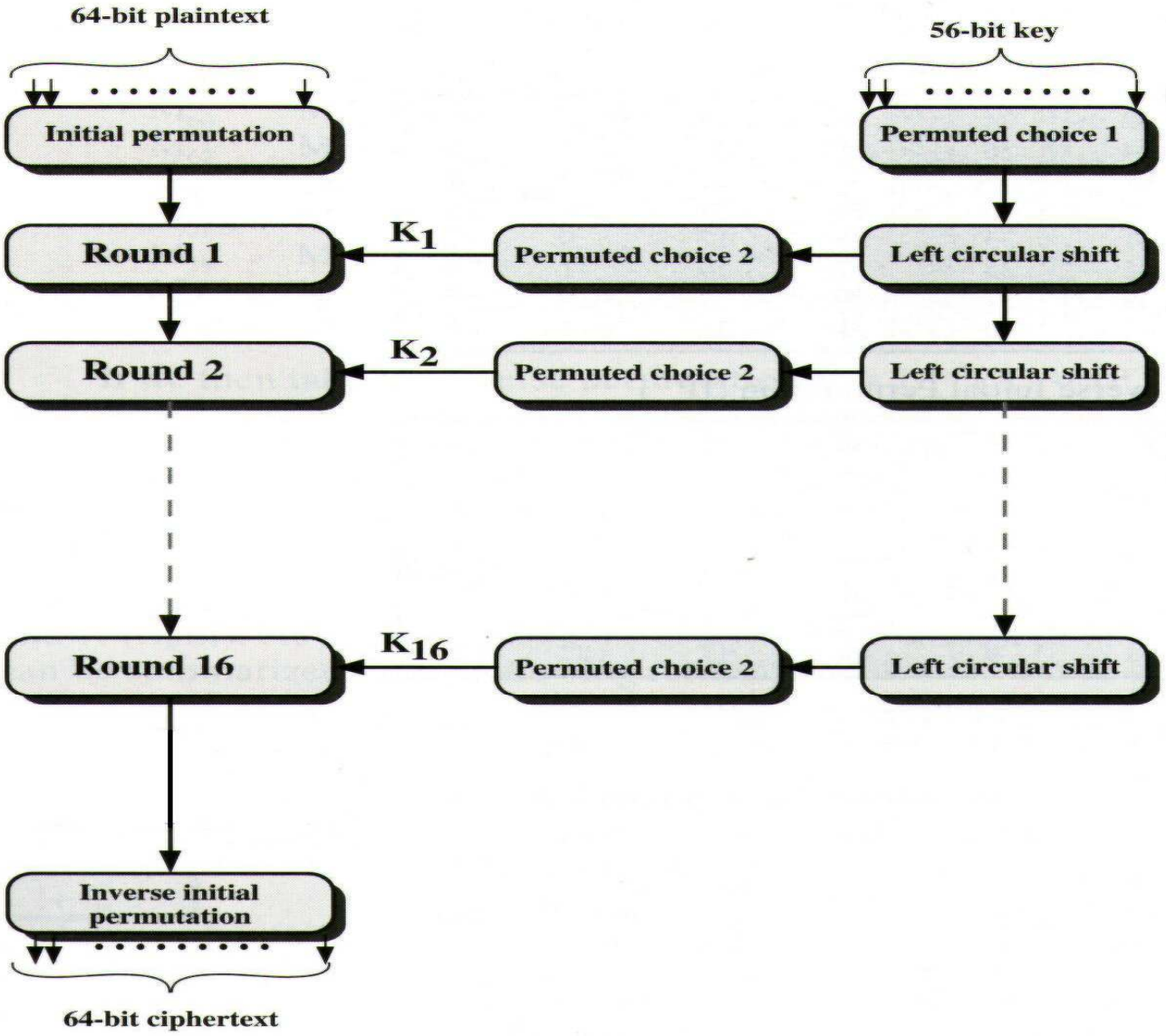
Key length in DES

- In the DES specification, the key length is 64 bit:
- 8 bytes; in each byte, the 8th bit is a parity-check bit



Each parity-check bit is the XOR of the previous 7 bits

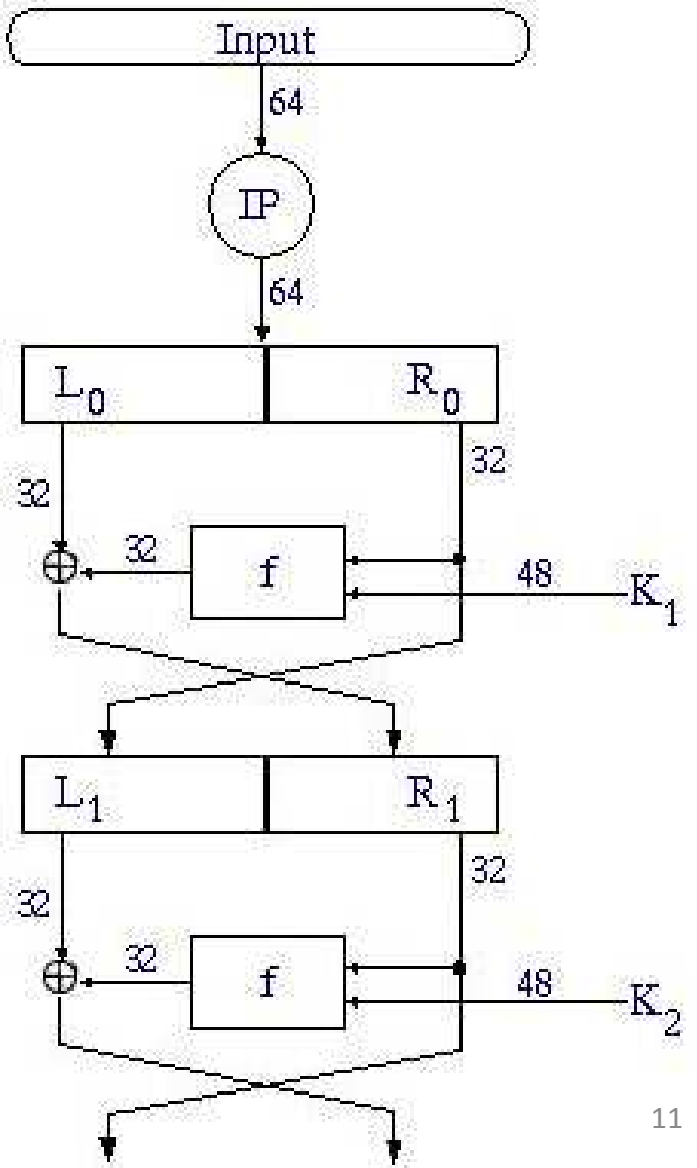
DES Rounds


64-bit plaintext
56-bit key
Initial permutation
Permuted choice 1
Round 1
K1
Permuted choice 2
Left circular shift
Round 2
K2
Permuted choice 2
Left circular shift
Round 16
K16
Permuted choice 2
Left circular shift
Inverse initial permutation
64-bit ciphertext

Details

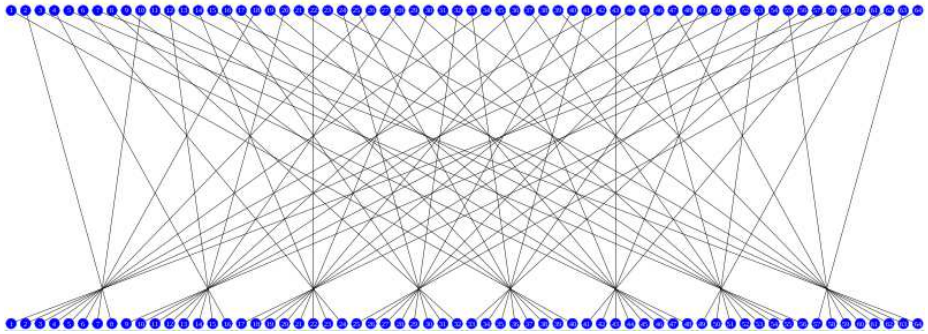
- $IP(x) = L_0R_0$
- $L_i = R_{i-1}$
- $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
- $y = IP^{-1}(R_{16}L_{16})$

Note: IP means Initial Permutation



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

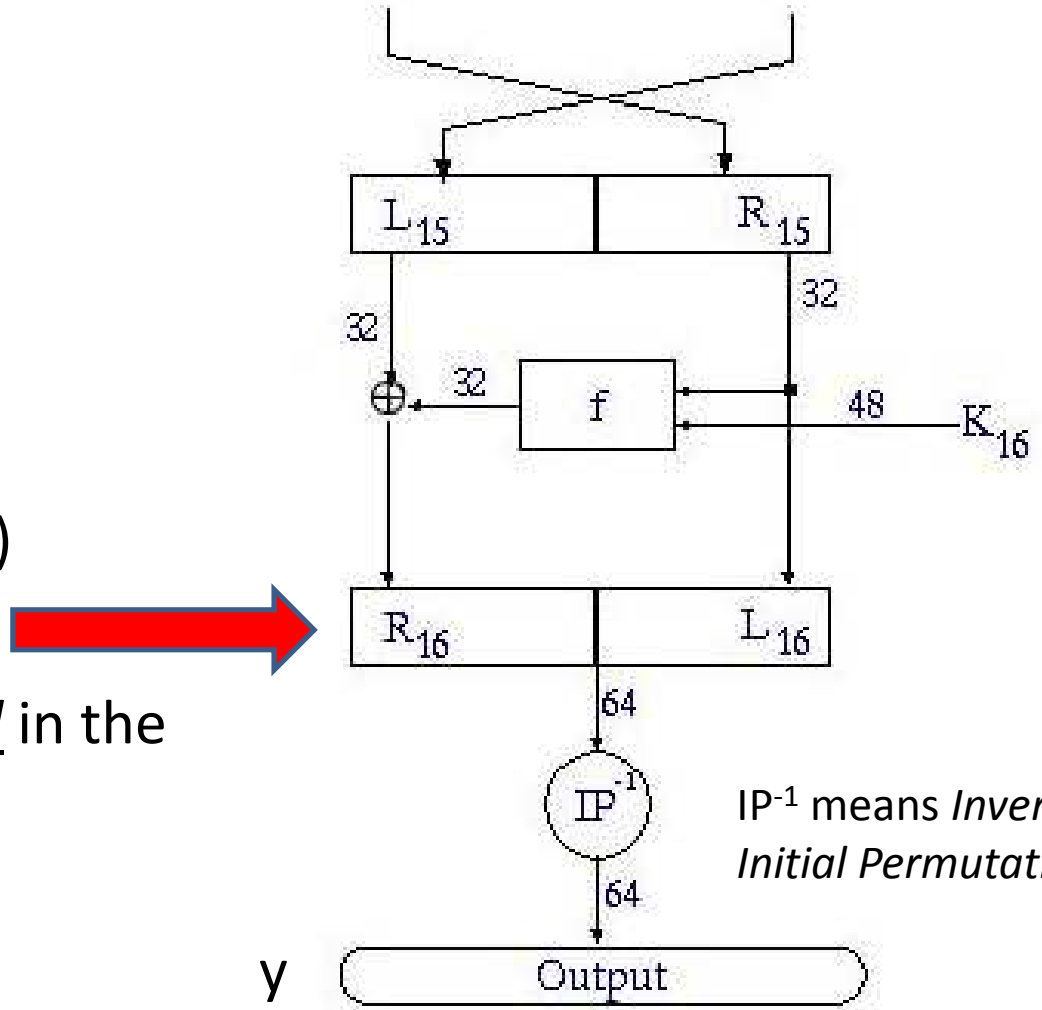


- This table specifies the input permutation on a 64-bit block.
- The meaning is as follows:
  - the first bit of the output is taken from the 58th bit of the input; the second bit from the 50th bit, and so on, with the last bit of the output taken from the 7th bit of the input.
- This information is presented as a table for ease of presentation:
  - it is a vector, not a matrix.

DES Rounds

- $IP(x) = L_0R_0$
- $L_i = R_{i-1}$
- $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
- $y = IP^{-1}(R_{16}L_{16})$

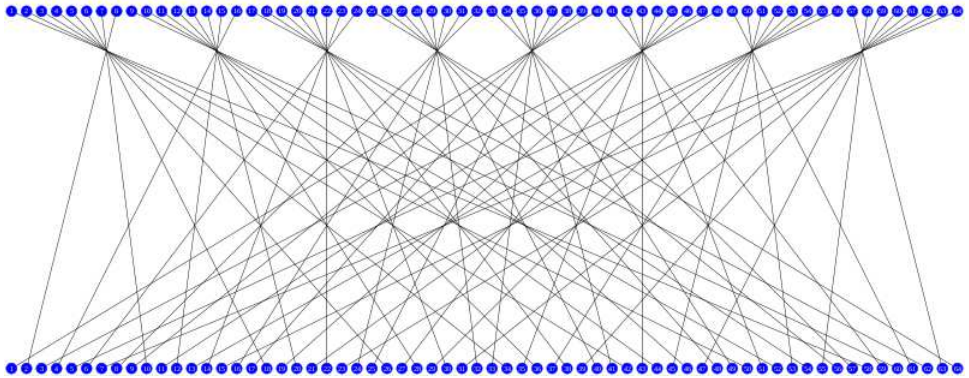
- Note that, as usual:
  - $R_{16} = L_{15} \oplus f(R_{15}, K_{16})$
  - $L_{16} = R_{15}$
- … but they are ***switched*** in the pre-output



IP^{-1} means *Inverse Initial Permutation*

Final Permutation (IP^{-1})

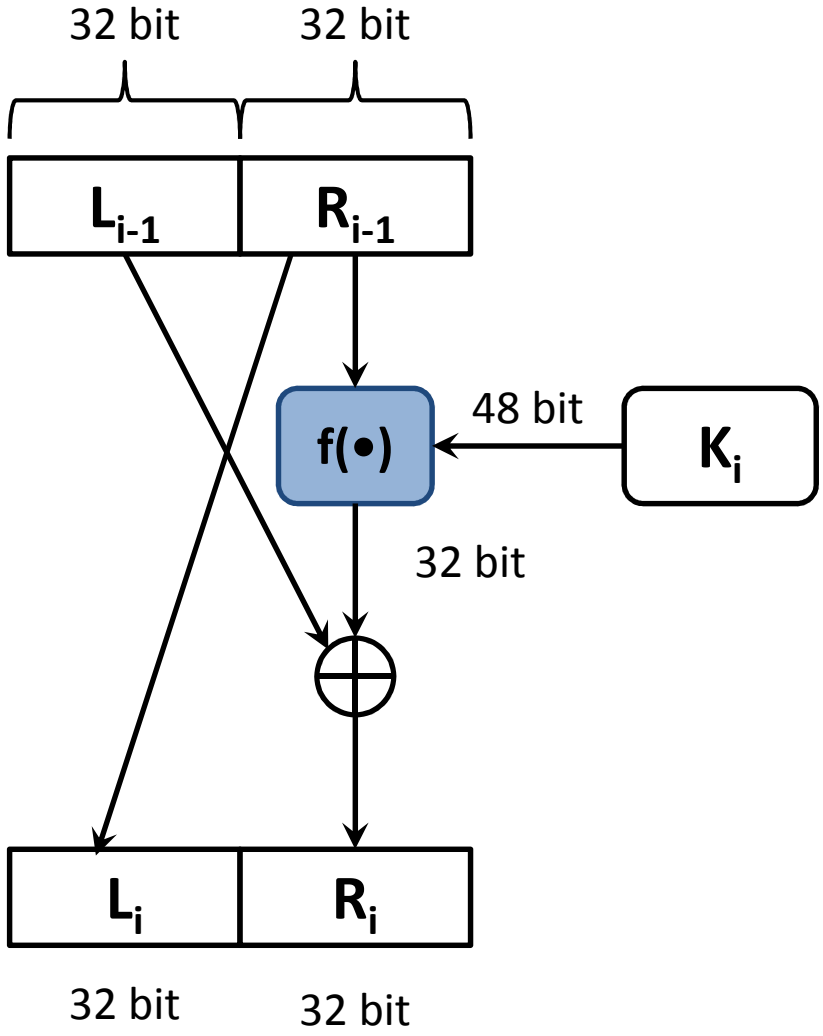
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25



- The final permutation is the *inverse* of the initial permutation; the table is interpreted similarly.
  - That is, the output of the *Final Permutation* has bit 40 of the preoutput block as its first bit, bit 8 as its second bit, and so on, until bit 25 of the preoutput block is the last bit of the output.

DES Round i

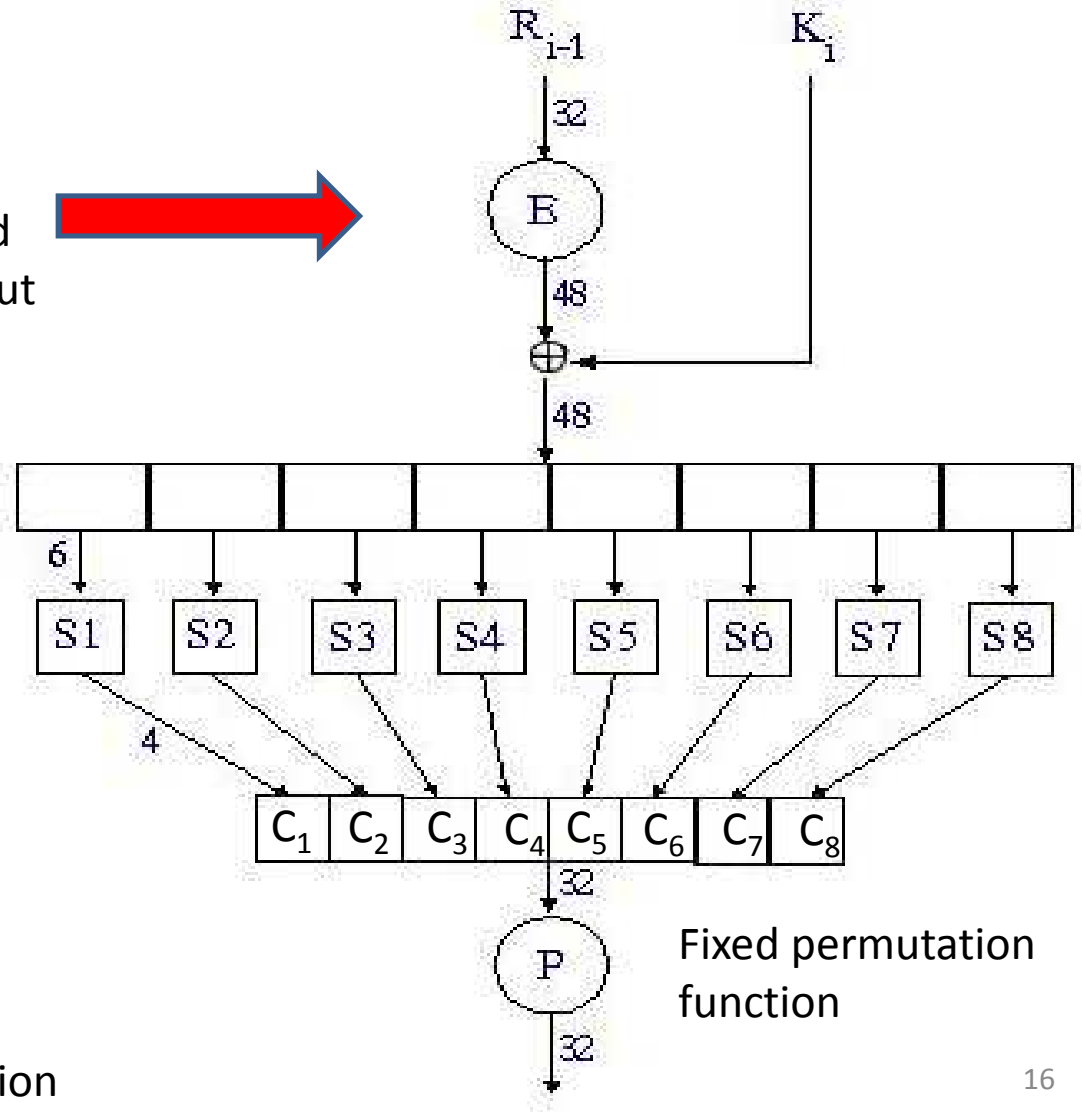
- $L_i = R_{i-1}$
- $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$



DES "f(•)" Function

E is an *expansion function* which takes a block of 32 bits as input and produces a block of 48 bits as output

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



Fixed permutation function

16 bits appear twice, in the expansion

S-boxes

- S-boxes are the only *non-linear* elements in DES design

Each of the unique selection functions $S_1, S_2, \ldots, S_8$, takes a 6-bit block as input and yields a 4-bit block as output

8 S-Box

- S = matrix 4x16, values from 0 to 15
- B (6 bit long) = $b_1 b_2 b_3 b_4 b_5 b_6$
  - $b_1 b_6$ ➔ r = row of the matrix (2 bits: 0,1,2,3)
  - $b_2 b_3 b_4 b_5$ ➔ c = column of the matrix (4 bits:0,1,…15)
- C (4 bit long) = Binary representation of S(r, c)

Example (S1)

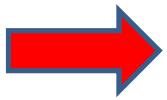
Row #	S_1	1	2	3	…	7	…	15	Column #
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S(i, j) <16, can be represented with 4 bits

Example: B =101111

b_1b_6 = 11 = row 3

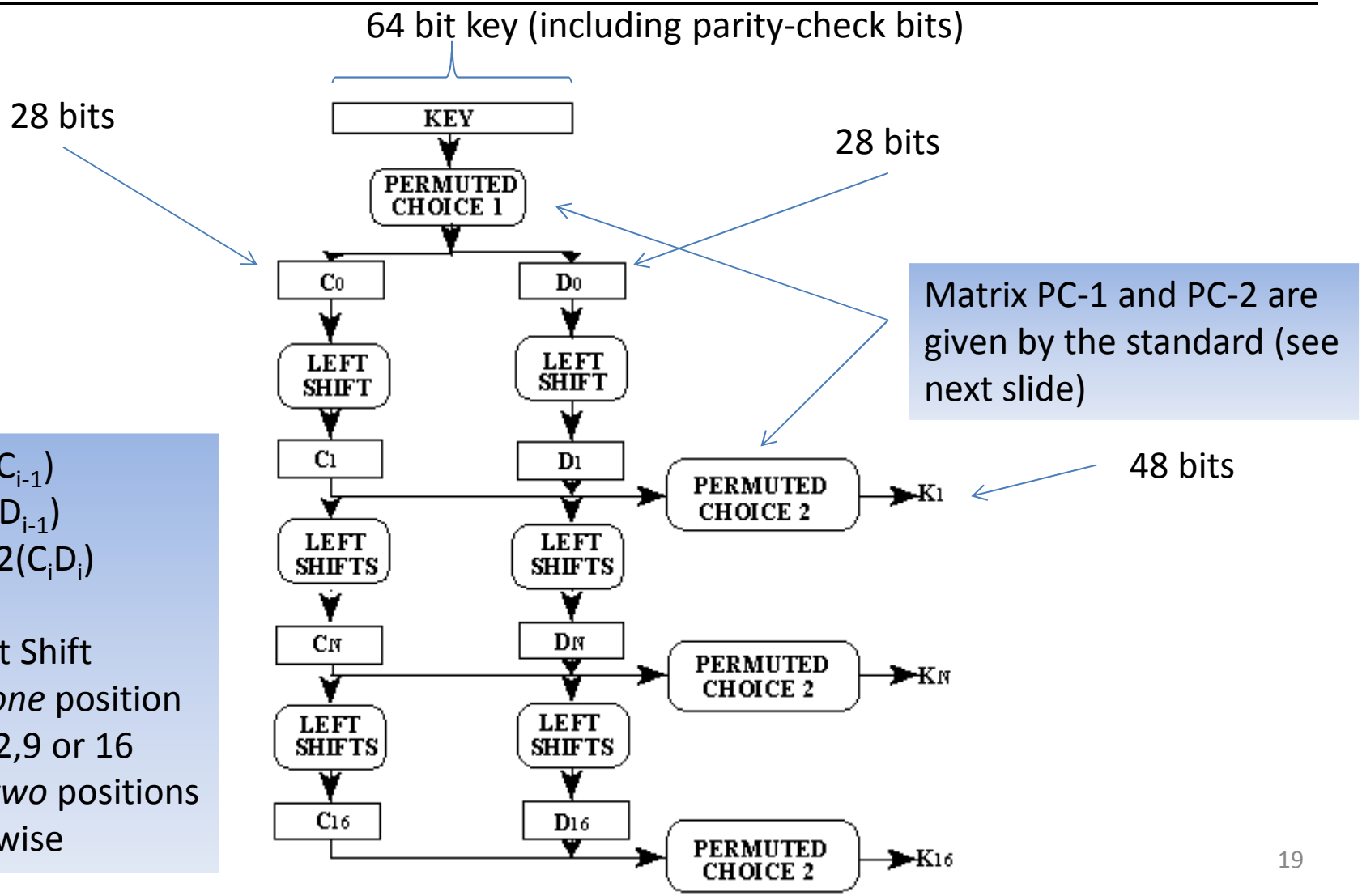
$b_2b_3b_4b_5$= 0111 = column 7



C=7=0111

Another example: B=011011, C=?

DES Key Generation (K_1 – K_{16})



DES Permuted Choice 1 and 2 (PC-1, PC-2)

Parity-check bits (namely, bits 8,16, 4,32,40,48,56,64) are not chosen, they do not appear in **PC-1**

Left						
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
Right						
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

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

PC-2 selects the 48-bit subkey for each round from the 56-bit key-schedule state

DES Weak Keys

- DES uses 16 48-bits keys generated from a master 56-bit key (64 bits if we consider also parity bits)
- Weak keys: keys make the same sub-key to be generated in more than one round.
- Result: reduce cipher complexity
- Weak keys can be avoided at key generation.
- DES has 4 weak keys
  - 01010101 01010101
  - FEFEFEFE FEFEFEFE
  - E0E0E0E0 F1F1F1F1
  - 1F1F1F1F 0E0E0E0E

DES Decryption

- Decryption uses the same algorithm as encryption, except that the subkeys K_1, K_2, …K_{16} are applied in reversed order

DES “f(•)” Function