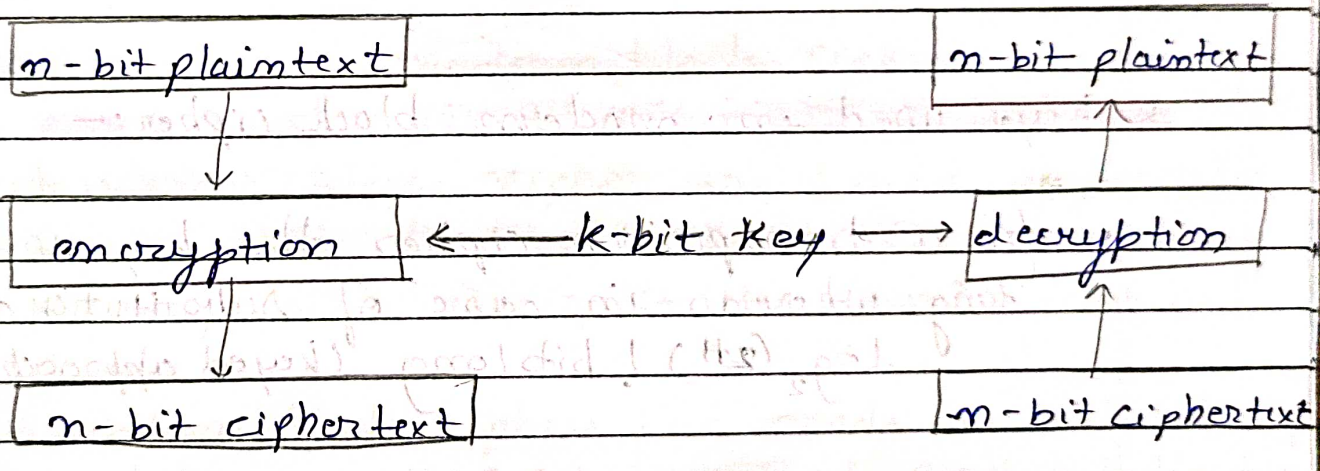
Page No.
Date / /

UNIT-2

* Modern block ciphers

- tradition ciphers methods are character oriented whereas modern ciphers methods are bits oriented where the bits are manage in blocks.
- following figure shows a modern block cipher where the plaintext is divided into n-bits blocks and if required padding bits are added to the last block.
- the value of n can be 64, 128, 256 or 512 bits

fig 5.1 A modern block cipher.



Q- how many padding bits must be added to a message of 100 character if 8 bit ASCII is used for encoding and block size should be 64 bits.

→ Since M = 100 characters and we are using 8 bits the size of message is = 100 × 8 = 800 bits.

Now, we have to convert 800 bits into 64 bit blocks.

no. of blocks ⇒ 12 ⇒ 64 bits

no. of padding bits ⇒ 32 + 32

0(B) P(B)

∴ total number of blocks required is 12 + 1 = 13 blocks

Page No.
Date / /

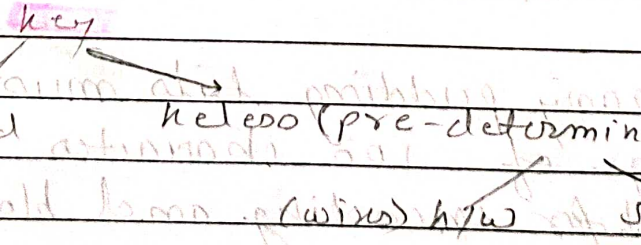
Q - A modern block cipher is design as a substitution cipher or transposition cipher justify your answer

→ Refer from ans chp 5

* keys used in modern block cipher:-

$n = 128, 64, 256$ or 512 bit

- for a transposition cipher the key is $\log_2 n!$ bit long whereas in case of substitution cipher is $\log_2 (2^n)!$ bits long (keyed approach)



- keyless ciphers are categories into two parts keyless transposition cipher and keyless ~~transposition~~ cipher substitution cipher.

← In keyless transposition cipher their is a fixed key size that consist of permutation rules and is represented as a table when

Page No.
Date / /

implemented in software.

- keyless transposition cipher are also called as P-box - permutation box. (reorder of the bits)

D-box - diffusion box or

* Keyless substitution cipher:-

- In these method a fixed key of a predetermined predefine mapping from the input to the output is design.
- mapping is define either in the form of a table or a mathematical equation.
- keyless substitution cipher is known as S-box {Substitution box} (mapping of the bits)

* components of modern block ciphers:- (generally the key can be choosen prime no)

- modern blocks ciphers are keyed substitution cipher in which the key allows partial mapping from the possible inputs to the possible outputs.

- modern blocks ciphers are made a combination of transposition units {P-box, D-box} and substitution units {S-box}

- → diffusion / confusion

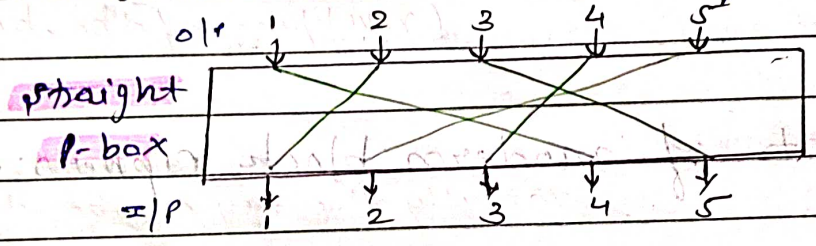
Page No.
Date / /

P-Boxes

* A P-box (permutation box) are of three types:-
  (a) straight P-box
  (b) compression P-box
  (c) expansion P-box

(a) straight P-box:-

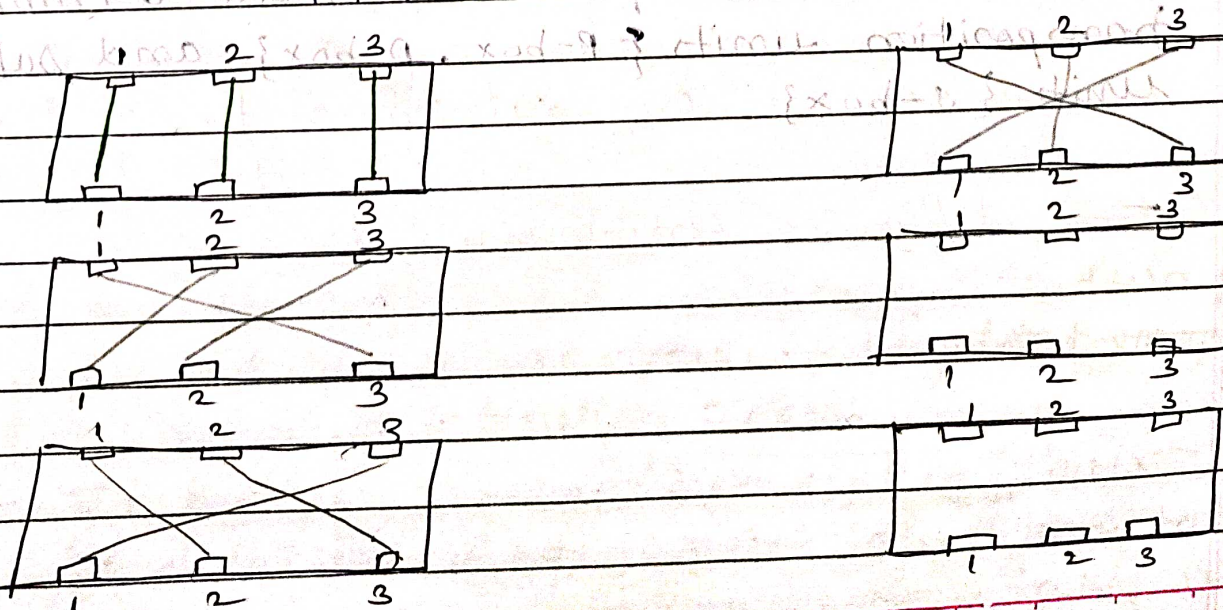
- In straight P-box it has 'n' inputs and 'n' outputs.
- Their are n! mapping possible, following figure shows the above concepts:-



2 Marks- Show all the possible mapping for 3×3 Pox :-
i/p o/p

→ n! = 3! = 3×2×1 = 6

total 6 combination must be their



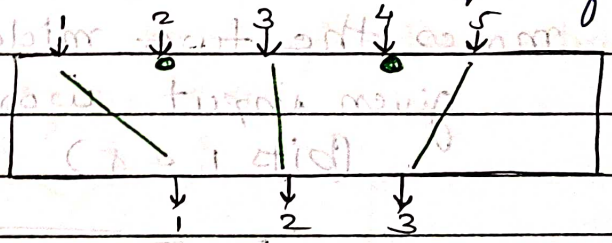
Page No.
Date / /

- P-boxes are key less were the mapping is already predetermined. The mapping can be either prewired (hardware) or either table (software)

- following shows the examples were the entries in the table are the input and the position of the entries are the output :-
  * from the given table the total blocks is 64 or the size of p-box is 64

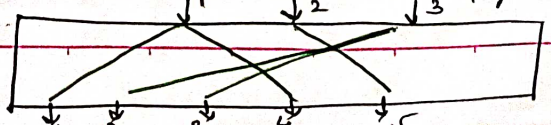
(b) Compression p-box :-
- In compression p-box it has 'n' - inputs and 'm' - outputs were $m < n$ that is some of the inputs are blocked and do not reach the output

- following figure shows an example of compression p-box :-



- Compression p-box are used when we need to permute the bit and at the same time decrease the number of bits for the next stage.

(c) expansion p-box :-
- An expansion p-box has 'n' inputs & 'm' outputs where 'm > n'.
- Thus, some of the outputs are connected to more than one inputs.
- following figures shows the expansion p-box :-



- expansion P-box are used when we need to permute the bits and at the same time increase the number of bits for the next step

Que-1] Design an 8×8 permutation table for a 'P' box that moves the two middle (bits 4 & 5) in the given input words to the two ends (bit 1 & 8) in the output word. relative position of the other bits should not be changed

→ - 8×8 permutation table for a 'P' box as m=n than it is straight 'P' box.

- As given :- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

- moves the two middle (bits 4 & 5) in the given input words to the two ends (bits 1 & 8) So, the output will be :-

| 4 | 2 | 3 | 1 | 8 | 6 | 7 | 5 |

| 4 | 1 | 2 | 3 | 6 | 7 | 8 | 5 |

Page No.
Date / /

Qu-19] A message has 2000 characters. if it is supposed to be encrypted using a block cipher of 64 bits, find the size of the padding and the number of blocks.

Given:-

→ message = 2000 characters — 8 bits ASCII
encrypted using block cipher = 64 bits. — 2000×8 = 16000

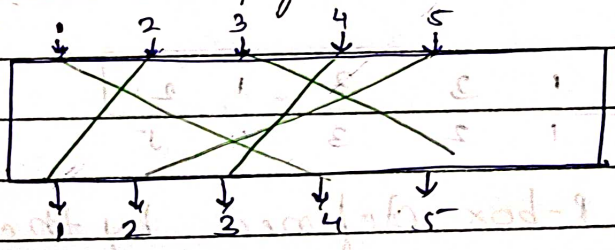
the size of padding bits are = 16000 % 64

64) 16000 (250

no. of blows are = 250

Qu-20] Show the permutation table for the straight P-box in figure 5.4

→ diagram



Iska input kaise ara '2' pos.

2	5	4	1	3
1	2	3	4	5

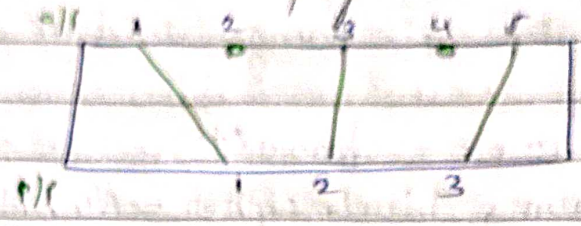
O/P →

Page No.

Date / /

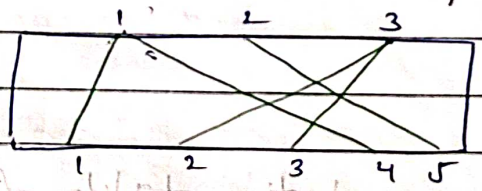
Que-21] Show the permutation table for the compression P-box in figure 5.4 :-

→



o/p → | 1 | 3 | 5 |
1 2 3

Que-22] Show the permutation table for the expansion P-box in figure 5.4 :-

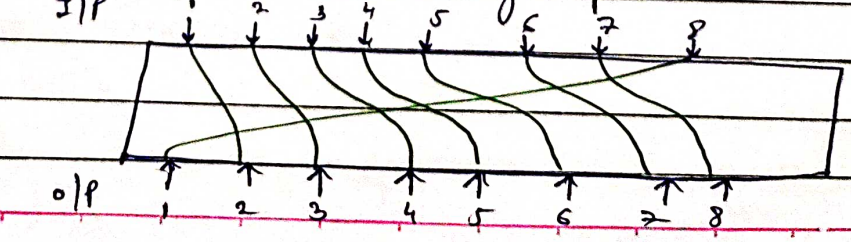


| 1 | 3 | 3 | 1 | 2 |
1 2 3 4 5

Que-23] Show the P-box defined by the following table:-

| 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

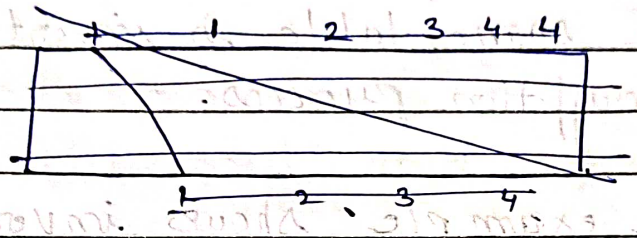
→ the above table shows 1-8 same range hence, it is straight P-box

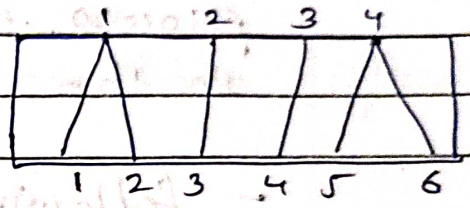


Page No.
Date / /

Qu-24] Determine whether the p-box with the following permutation table is a straight p-box, a compression p-box, or an expansion p-box. [1 1 2 3 4 4]

→ As the number of inputs are 6 and output is 4 i.e. 6×4 hence, it is a expansion p-box.





Qu-25]

Page No.
Date / /

* Invertibility :-

- from the three types of P-boxes, a straight P-box is invertible because the number inputs and outputs are mapped 1:1.

- Thus, permutation table used in the encryption process, the same table is inverted and used in the decryption process.

- following example shows inverting a given straight P-box permutation table :-

Que

1) original table

6	3	4	5	2	1

6	3	4	5	2	1	2) Add indices
1	2	3	4	5	6	

3) Swap content & indices

1	2	3	4	5	6
6	3	4	5	2	1

4) Sort the table

6	5	2	3	4	1
1	2	3	4	5	6

pg:- 158 of 752 fig 5.7

Que - A straight P-box is invertible, but compression and expansion P-box are not. Why? Justify or explain with the example.

Page No.
Date / /

* Substitution Box or S-Box:-

- Substitution Box or S-Box can have different number of input and output.

- S-box can be keyed or keyless but most modern block cipher normally used keyless where the mapping is predetermined.

- S-box is an 'm x n' substitution unit where m and n are not same. 'm x n' ⇒ # ≠

- S-boxes are represent using matrices for the given mathematical equations.

- S-boxes are classified as linear s-box and non-linear s-box.

- linear versus non-linear s-boxes :- In an s-box with 'n'-inputs and 'm' outputs, we call the inputs $x_0, x_1, \ldots, x_n$ and the outputs $y_1, \ldots, y_m$. the relationship between the inputs and the outputs can be represented as a set of equations.

$$y_1 = f_1(x_1, x_2, \ldots, x_n)$$
$$y_2 = f_2(x_1, x_2, \ldots, x_n)$$
$$\vdots$$
$$y_m = f_m(x_1, x_2, \ldots, x_n)$$

Page No.
Date / /

- In a linear s-box, the above relations can be expressed as:-

$y_1 = a_{1,1}x_1 \oplus a_{1,2}x_1 \oplus \cdots \oplus a_{1,n}x_n$

$y_2 = a_{2,1}x_1 \oplus a_{2,2}x_1 \oplus \cdots \oplus a_{2,n}x_n$

$y_m = a_{m,1}x_1 \oplus a_{m,2}x_1 \oplus \cdots \oplus a_{m,n}x_n$

- In a non-linear s-box we cannot have the above relations for every output the terms in non-linear s-box will be represented as -

$x_1 \cdot x_2$, $x_3 \cdot x_5$ etc

Qu- for the given equation $y_1 = x_1 \oplus x_2 \oplus x_3$, $y_2 = x_1$. find weather the s-box is linear or non-linear. Represent the relationship using matrices.

⟶ given:- $y_1 = x_1 \oplus x_2 \oplus x_3$, $y = x_1$

Soln:- the above given equations show that it is a linear equation. s-box.

$y_1 = a_{1,1}x_1 \oplus a_{1,2}x_2 \oplus a_{1,3}x_3$

$y_2 = a_{2,1}x_1 \oplus \underset{0}{\underline{\quad}} \oplus \underset{0}{\underline{\quad}}$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Page No.
Date / /

Qu- In an s-box with three inputs and two outputs, we have the equations as, $y_1 = x_1 \cdot x_2$ $y_2 = x_1 + x_2 x_3$ identify the type of the s-box.

→ given:- $y_1 = x_1 \cdot x_2$ $y_2 = x_1 + x_2 x_3$

Soln:- the above equation is a non-linear s-box as the it is non-linear because their is no relationship between the i/p and o/p hence, the matrix can't be drawn.

Qu- s-box may or maynot be invertible justify your answer.

→ * for an invertible s-box the number of inputs bits should be equal with the number of outputs bits following shows an examples.

* I/o relation of 2x2 s-box is shown in the following table:-

left most ↓ \ right most →	0	1
0	01	11
1	10	00

* show the inverse s-box table:

- As the no. of input bits = output bits hence, the inverse can be taken out.

Soln:- {for the above table the left most bit of the given input defines the row and the remaing rightmost bits define the coloum.}

- The above table can be represented as :-

I/P ⇒	00	01	10	11
o/p ⇒	01	11	10	00
Add	1 (0)	3 (1)	2 (2)	0 (3)
swap	0	1	2	3
sort	3 (1)	0 (3)	2 (2)	1 (3)

Page No.
Date / /

O/P :- 3 0 2 1
11 00 10 01

I/P :- 00 01 10 11
O/P :- 11 00 10 01

	0	1
0	11	00
1	10	01

⇒ Invertible S-box

Qu- find the invertible 3x3 S-box for the given values :-

	00	01	10	11
0	011	101	111	100
1	000	010	001	110

→

I/P :⇒	000	001	010	011	100	101	110	111
O/P :⇒	011	101	111	100	000	010	001	110
	3	5	7	4	0	2	1	6

Add indices

3	5	7	4	0	2	1	6
0	1	2	3	4	5	6	7

Swap indices

0	1	2	3	4	5	6	7
3	5	7	4	0	2	1	6

Sort

4	6	5	0	3	1	7	2
0	1	2	3	4	5	6	7

100 110 101 000 011 001 111 010

Page No.

Date ___/___/___

O/P 100 110 101 000 011 001 111 010

I/P 000 001 010 011 100 101 110 111

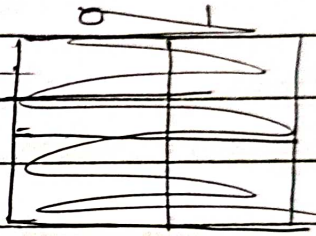
	00	01	10	11
0	100	110	101	000
1	011	001	111	010

⇒ Invertible S-box

Ques-

	00	01	10	11
0	110	001	010	011
1	100	111	101	000

Ques-



Ques-

	0	1
0	00	10
1	11	01

Que- write the short note third component (Ex-OR operation)

② other component :- { x-or operation, complement, Inverse, circular shift (L & R), swap, split & combine }

1) P-box 2) D-box 3) S-box 4) Linear Non-Linear.

Que-15] (a) Show the result of 3-bit circular left shift on word $(10011011)_2$

(b) Show the result of 3-bit circular shift on the word resulting from part a.

(c) Compare the result of part b with the original word in part a.

→ (a) Soln :- find :- result of 3-bit circular leftshift

given :- $(\overset{b_7}{1}\ \overset{b_6}{0}\ \overset{b_5}{0}\ \overset{b_4}{1}\ \overset{b_3}{1}\ \overset{b_2}{0}\ \overset{b_1}{1}\ \overset{b_0}{1})_2$

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	0	0	1	1	0	1	1
1	1	0	1	1	0	0	1

(b) Soln :- find :- result of 3 bit circular right shift

given :- 1 1 0 1 1 0 0 1

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	1	0	1	1	0	0	1
1	0	0	1	1	0	1	1

(c) Soln :- compare the result of part b & part a.

	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
a ⇒	1	1	0	1	1	0	0	1
b ⇒	1	0	0	1	1	0	1	1

Q16, 17

Page No.
Date __/__/

Q.11-16] a) Swap the word $(10011011)_2$

b) Swap the word resulting from part a

c) Compare the result of part a and part b to show that swapping is a self-invertible operation.

→

(a) Soln :- $(10011011)_2$

0 0	0
0 1	0
1 1	0
1 0	1

Page No.
Date / /

Q.11-17) find the result of the following operations:-

(a) (01001101) ⊕ (01001101) ⇒ 00000000

(b) (01001101) ⊕ (10110010) ⇒ 11111111

(c) (01001101) ⊕ (00000000) ⇒ 01001101

(d) (01001101) ⊕ (11111111) ⇒ 10110010

Page No.
Date / /

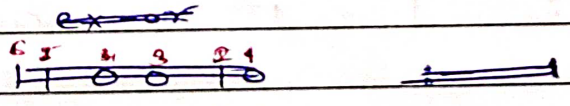
Qu- A 6x2 s-box is exored the odd number bits to get the left bit of the output. similarly perform ex-or operation on the even number bits to get the right bit of the output. perform the mentioned on the following input string.

1) 110010 2) 101101

→ given:- • 6x2 s-box

find:- • odd number bits to get the left bit of o/p.
• even number bits to get the right bit of o/p.

- odd number bits

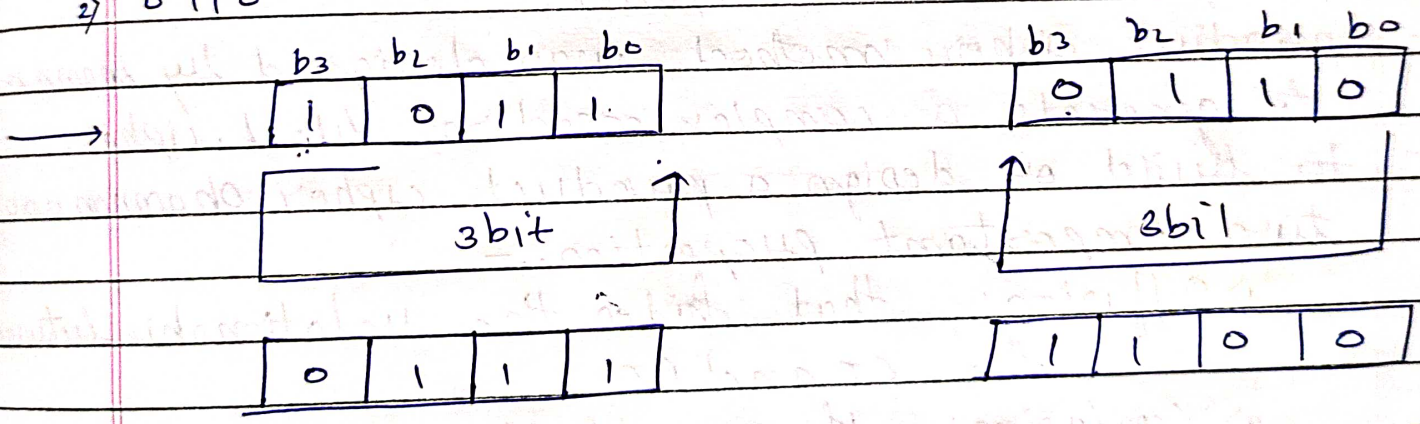


	6	5	4	3	2	1
	1	1	0	0	-1	0
1						
0						

Page No.
Date / /

Que- The left most bit of a 4x3 s-box rotates the other three bits if the left most bits is 0 then the other bits are rotated to right one bits similarly if the left most bits is one then the three other bits are rotated to the left one bit, perform the above and generate the output for the given inputs.

1) 1011
2) 0110



Q- Write short note on product cipher

Page No.
Date / /

* Product cipher :→ modern block cipher
  - It was introduced by Shannon
  ↓
  Substitution
  - Transportation
  - other components

* properties :-
  → Diffusion — hides the relationship of CT & PT
  → Confusion — confuse the relationship between key & CT

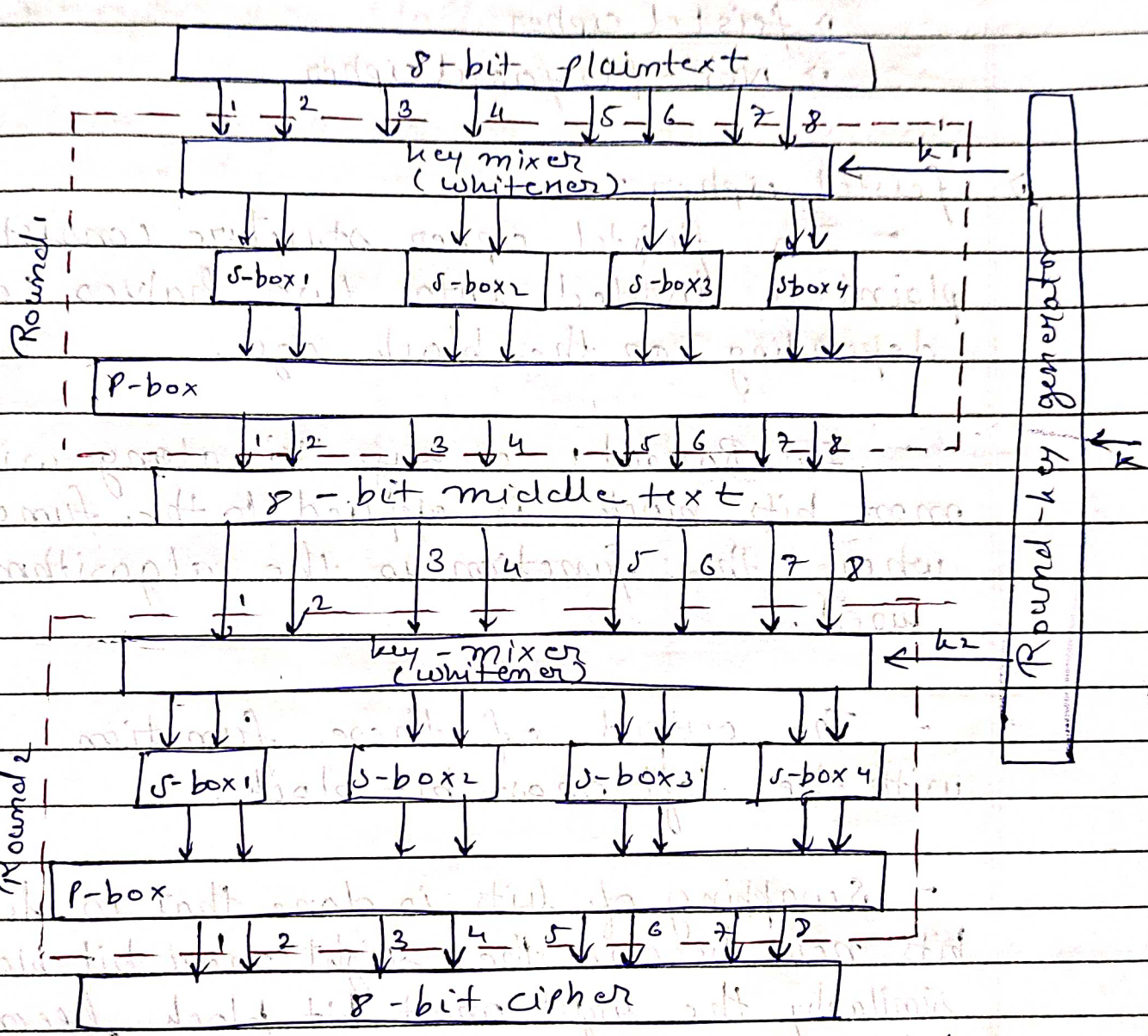
- Product cipher is the combination of substitution, transposition, other components.
- product ciphers method was designed by Shannon to generate a complex modern block cipher.
- to build or design a product cipher Shannon uses two important properties :-
  i) Diffusion :- that hides the relationship between the CT and PT
  ii) Confusion :- these properties confuses the attacker to find the key and CT.

- product cipher consists of 3 steps blocks :-
- ~~following figure shows~~
  Confusion and diffusion properties are applied number of time or iterated number of times that each iteration knowns as rounds.

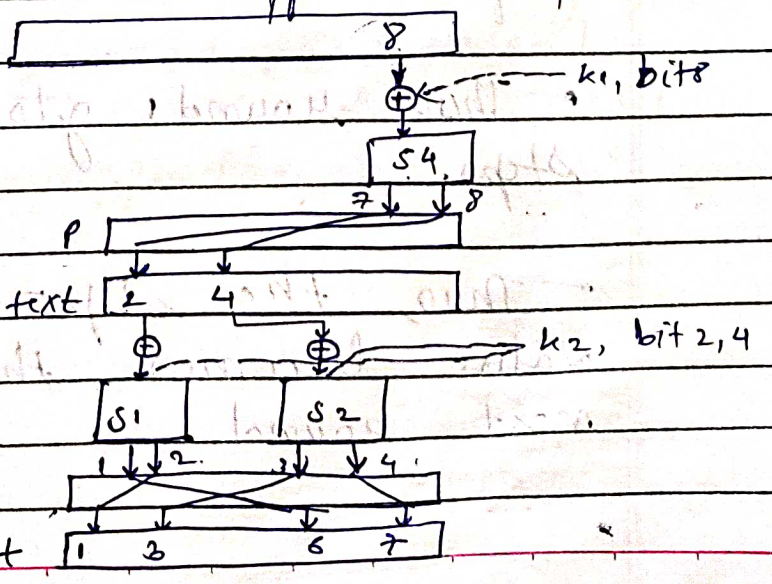
Page No.

Date / /

- following figure shows the product cipher of 2 rounds:-



- following figure shows the diffusion & confusion in a block cipher:-



Page No.
Date / /

* ~~Classes of~~ classes of product cipher :-
- product cipher are divided into two classes
  - ⇒ feistel cipher
  - ⇒ Non - ~~fiestel~~ feistel cipher

⇒ feistel cipher :-
- In feistel cipher structure consist of the plaintext divided into two halves equally depending on the block size.

- In Round 1, a key 'ki' along with right most bits block is applied to the function where the function is the algorithm to be used.

- The output of these function is exored with the left most bit block.

- Swapping of bits is done that is left most bits now become the right most bit block and similarly the right most bit block becomes the left most bit block

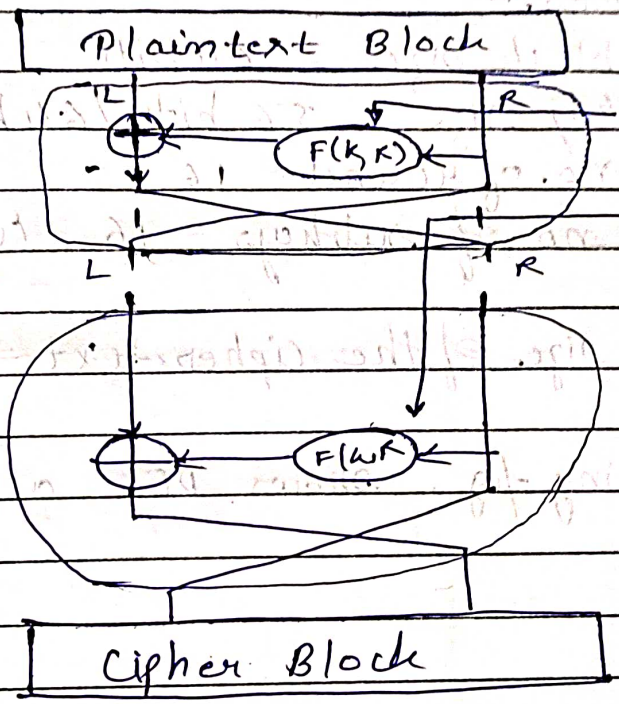
- Thus, round 1 gets completed in the above four steps.

- Thus, the step - 4 along with round key value becomes the next input for the next round.

Page No.
Date / /

* - Thus, after the number of rounds completed depending on the type of algorithm we get the final cipher text block.

- following figure shows the above process:-



* Block cipher design principles are based on the following:

1) Block size :- length of PT = CT [as per the algo]
2) key size :- master keysize = . [as per the algo]
3) total number of rounds :- conversion from PT → CT
4) No. of subkeys count :- as per the algo
5) Round function defination :

Page No.
Date / /

* DES Algorithm :- (Data encryption standard algo)

- DES algorithm is based on the concept of feistel structures.

- Thus, DES algorithm is based on the following principles:-
  - block size = 64 bits
  - key size = 56 bits / 64 bits
  - no. of round = 16
  - no. of subkeys = 16 (where each subkey size = 48 bits)
  - Size of the cipher text = 64 bits.

- following fig. shows DES algorithm process:-

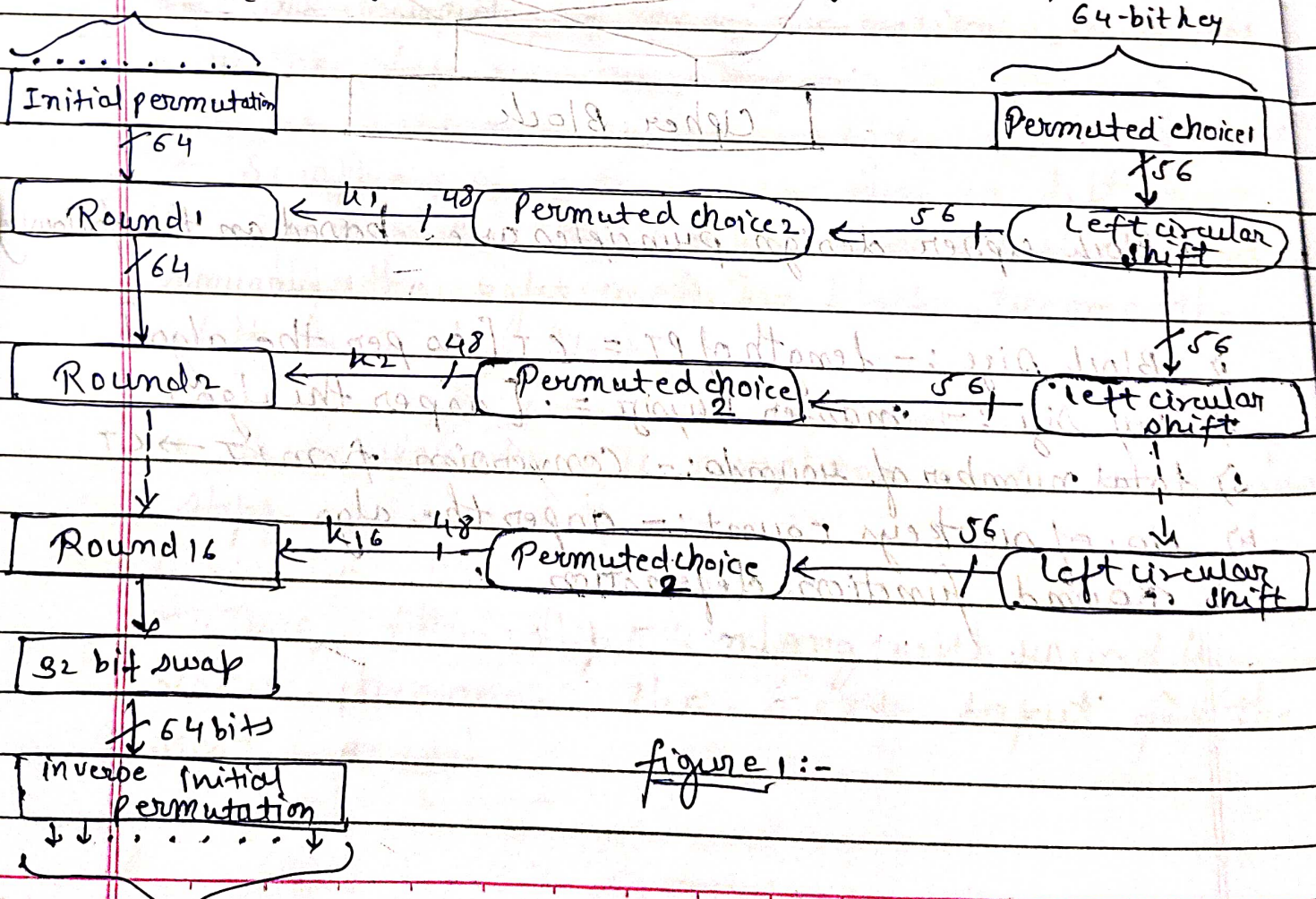


figure 1:-

Page No.
Date / /

- following fig. shows DES single round function process:—

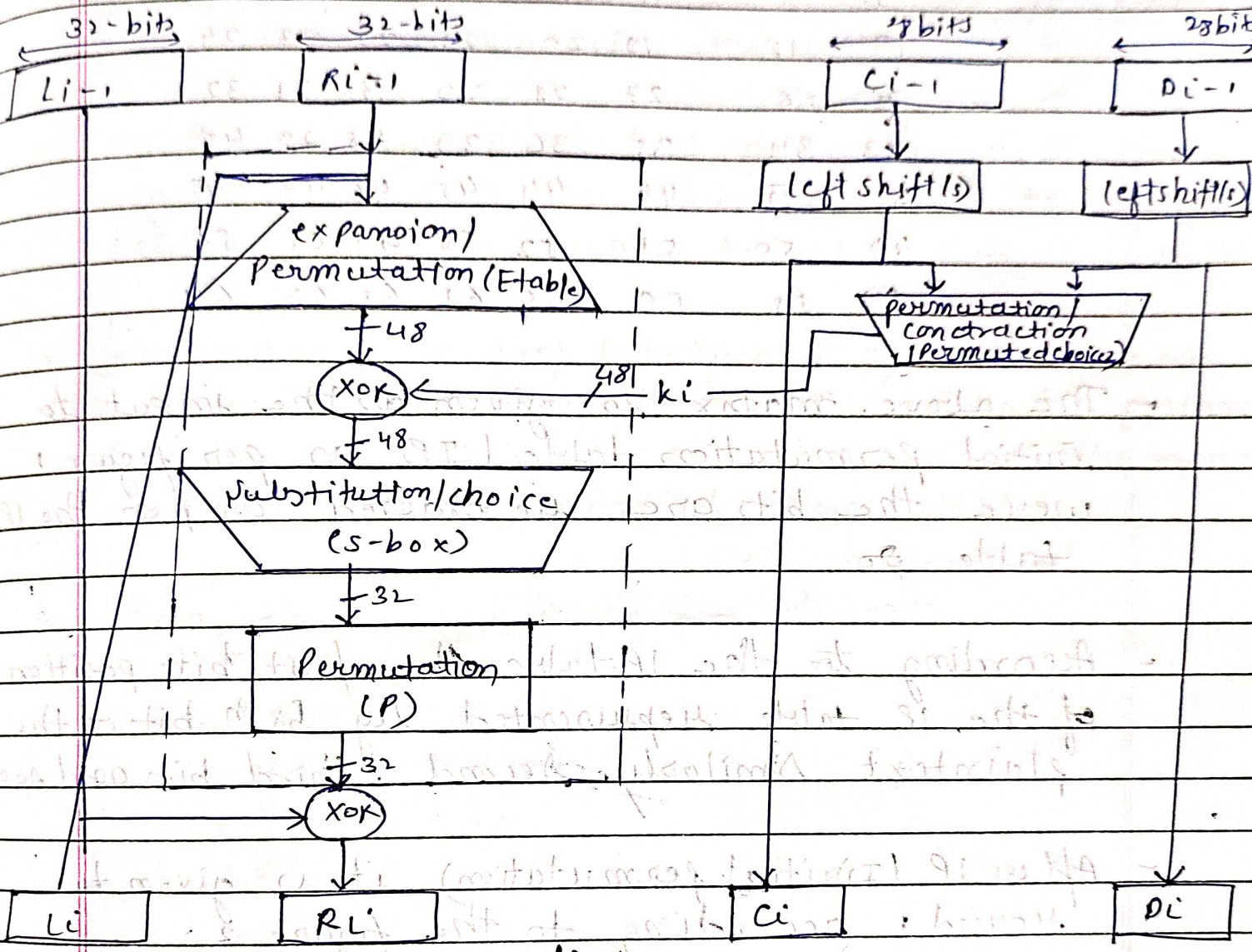


figure2:-

* DES encryption process:—

[step1] Consider the 64 bit plaintext as an input because in DES algorithm the block size = 64 bit. the 64 bit plaintext is represented as 8x8 matrix as shown below:

Page No.

Date / /

- 64-bit PT as 8×8 matrix form

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

- The above matrix is given as the input to initial permutation table (IP) as per figure 1 were the bits are re-ordered as per the IP table. So

- According to the IP table the first bit position of the IP table represented by 58th bit of the plaintext similarly second ; third bits and so on.

- After IP (Initial permutation) it is given to round 1 according to the figure 2.

- Thus, the plaintext is now divided into two equal halves each of 32 bit block size.

- Now, the right half 32 bits are given to the expansion permutation table where the bits are expanded from 32 → 48 bits.

Page No.
Date / /

- thus, the output of the expansion permutation table (EP-table) is need to be exord with the output of permutate choice 2 table for these we have to generate the subkey.

- As the key size = 64bit but the cipher key use for DES algorithm is 56 bit key thus, we need to apply permutated choice 1 table to reduce from 64 bit to 56 bit

- here, parity bit drop technique is applied where we need to discard 8 bits that is the parity bits positions from the given 64 bit key and then permute the rest of the bit using permutated choice, table.

- thus, the bit positions — discards.
  8th bit = parity bits
  16th bit = —"—
  24th bit = —"—
  32th bit = —"—
  40th bit = —"—
  48th bit = —"—
  56th bit = —"—

* 57th bit of the key is the first position as a output

- according to the figure two 56 bits are divided into two equal halfs each of 28 bits.

- Thus, each 28 bits is given as input to LCS. (left circular shift). thus, number of bits to be shifted depends on the round number. refer the table where for

Page No.
Date / /

Round 1 the number of bit to be shifted is equal to 1, refer the table for more details.

- The output of LCS is given to permutate choice 2 table where reordering of bits are performed and the bits are reduce from 56 to 48 bits
  - { The discards bits are (9, 18, 22, 25, 35, 38, 43, 54) } these is also called as contraction & reduction method.
- The output of permuted choice 2 table is with ex-ored with expansion permutation table (properties is closure)
- the 48 bit output of ex-ored result to be applied to s-boxes. were, the bits are reduced from 48 to 32 bit

row → 00 - 0 10 = 2nd row
01 - 1 0011 = 3rd colum
10 - 2 ∴ 1000
11 - 3

column → 0000 = 0
0001 = 1
1111 = 15

8×4 = 32
⇒ row 0 = 2bit = 4 rows.

S1 → [1] 0 0 [1] [1] [0]
column = 4 = 2^4 = 16 colum.

8 × (6) = 48 6 bit

S1 ... S8

reduced it to 8 × (4) = 32bit

Page No.
Date / /

- So the 6th bit input 100110 is reduced to 1000 that is equal to 4 bit thus the process is repeated for total 8 s-box tables to get 32-bit output.

- Now, the output of s-box that is 32 bit is given to the permutation function P where the bits are again rearranged according to the following table.

- The output of p-function of 32 bit is Ex-or with left (32 bit) most bit and stored in right most bit.

- Similarly the right half will be stored with left half and these will be act as a input for the next round along with the key values.

- The process is repeated for 16 number of rounds. After the 16th round 32 swap is there and is applied to inverse initial permutation table denoted as I(p-1) (refer the table from the slide)

- Thus finally, the 64 bit or output of IP inverse is generated as cipher text.

Qu - Write the explanation for single round permutation
2M Qu - What are the bit discard in permuted choice two
Qu - 6bit to 4bit (in S-box)

Self studies topic.

Page No.

Date / /

Que - The input to s-box1 is 100011 what is output

→

[1] 0 0 0 1 [1]

unit colm

row = 11

column = 0001

(12)

- it is called as s-box rule.

Que - write the s-box rule.

Pg167 Table 6.10 - S-boxes

que - if the given input is 000000

→

que - ii) Answer the following que about S-box in DES :-

a) Show the result of passing 110111 through S-box 3

→ row = 11 = 3 column = 1011 = 11

(3)

b) Show the result of passing 001100 through S-box 4

→ row = 00 = 0 column = 0110 = 6

(5)

c) show the result of passing 000000 through S-box 2

→ row = 00 column = 0 (4)

d) show the result of passing 111111 through S-box 2

→ (2) row = 11 = 3 column = 15

Page No.
Date / /

* Blow fish algorithm:-

- blow fish is a symmetric key algorithm (same key is use to perform encry & decry) ~~the si~~
- the size of the plain text is 64 bit
- it uses the concept of feistel structure
- it has variable key size as compare to DES (56bit) (32bit to 448bit)
- blow fish was developed by Bruce schiener in 1993
- total number of rounds = 16 to convert P.T to C.T
- it is alternative to DES algorithm method.
- blow fish algorithm is divided into two parts :-
  1) generation of sub keys.
  2) Data encryption / decryption process.

* generation of sub keys:-

- As the key size is variable that is of the size 32 bit to 448 bit which is divided into 14 sub keys i.e $k_1, k_2, \ldots k_{14}$ $1 \le n \le 14$ each of 32 bit size

$\frac{448}{32} \Rightarrow 14$

- blow fish introduces the concept of p-array and s-boxes.
- thus, their are total 18 p-array numbered from P_1 to P_{18} or P_0 to P_{17} followed 4 s-boxes of size 8×32

- each s-box has total 256 enteries (0 to 255) which is defined as $S_{1,0}, S_{1,1}, \ldots S_{1,255}$

$S_{2,0}, S_{2,1}, \ldots S_{2,255}$

$S_{3,0}, S_{3,1}, \ldots S_{3,255}$

$S_{4,0}, S_{4,1}, \ldots S_{4,255}$

$S_{1,0}$ → 0 entry
↓
s-box 1
→ 255 entry
$S_{1,255}$
↓
s-box 1

- each s-box has 8 bit input and 32 bit output thus in 8-bit input 1st, 7th, 8th ⇒ represents the row no. where as the remaining bit (2, 3, 4, 5, 6) ⇒ represents the column no.

- thus, the interaction of row and column determine the value of s-box.

* initialize the p-array value first followed by initializing four s-boxes with a fixed string where the values are the bits representing the fractional part of the pi (π) value in hexadecimal form.

Generating sub key

** following shows some initials value of sub keys:-

P[1] = 243f6a88

P[2] = 85a308d3

...

P[18] = 21c638c4

...

S[4,255] = 8ac3 7236

Page No.

Date / /

Perform bit wise ex-or operation between the P-array and key values: assuming the key value as 448 bits.

$k = \frac{448}{32} = 14$

$k_1 - k_{14}$

$P_1 - P_{18}$

→

$P_1 = P_1$ X-or k_1 ( )

$P_2 = P_2$ X-or k_2 ( )

|

$P_{14} = P_{14}$ X-or k_{14} ( )

$P_{15} = P_{15}$ X-or k_1 ( )

$P_{16} = P_{16}$ X-or k_2 ( )

$P_{17} = P_{17}$ X-or k_3 ( )

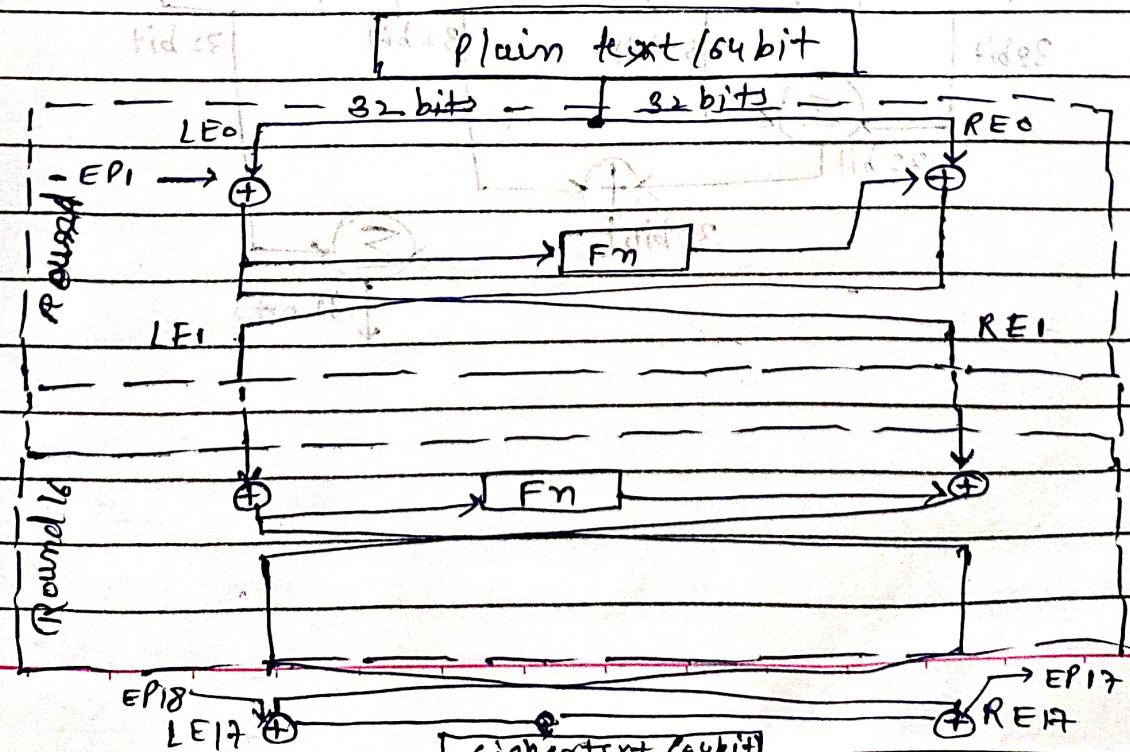
$P_{18} = P_{18}$ X-or k_4 ( )

Reuse of key

$k_1 \leftrightarrow k_4$

* Blowfish encryptim process

In these process a 64 bit plain text is divided into two equal half and with the use of P-array and S-boxes in the function F following shows the blow fish encryption diagram.



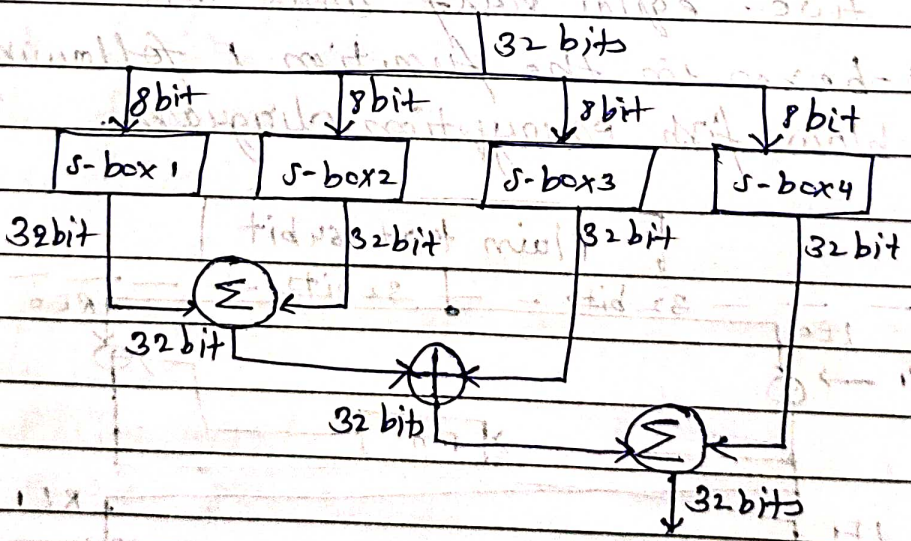
Page No.
Date / /

- Divide the given PT of 64 bit two equal half L & R of 32 bit each.

- ~~Algorithm~~: then the :-
  for i=0 to 16 {
  $L_i = L \oplus P_i$
  Function (L_i)
  $R_i = fun(L_i) \oplus P_i$
  }swap (L_i, R_i)

  unswap (L_i, R_i)
  then,
  $L = L \oplus P_{18}$
  $R = R \oplus P_{17}$
  we get cipher text.

- the function F in the above figure is as follows: -



Page No.
Date / /

The function f in fig A is as follows as shown in fig (B).

① Divide the 32-bit left block into four 8 bit subblocks i.e 8bit s blocks named as f[a, b, c, d]

② Compute

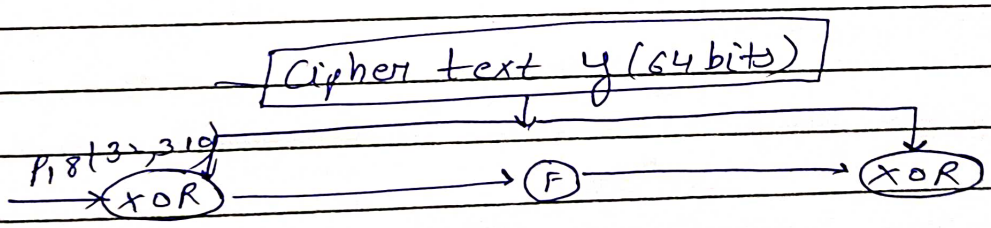
$f[a, b, c, d] = ((S_1, a + S_2, b) \text{ XOR } S_3, c) + S_4, d.$

let a = 10, b = 95, c = 37, d = 191

Then $f[a, b, c, d] = ((S_1, 10 + S_2, 95) \text{ XOR } S_3, 32 + S_4, 191)$

following fig. shows the blowfish decryption process.

Actual Phase isopose Chp-3 3rd edition



Page No.

Date / /

Q Justify how to generate the sub-key values using blowfish encryption algo.

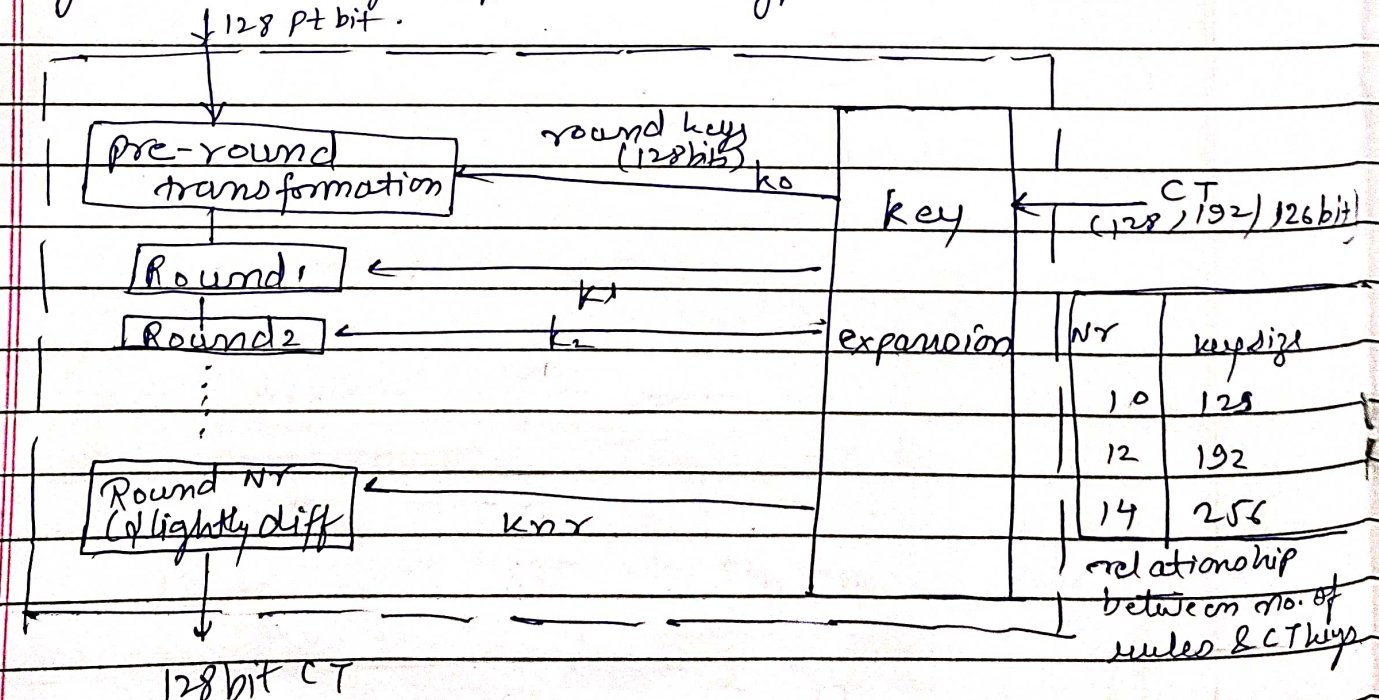
→

Page No.
Date / /

* AES Algorithm (Advance encryption cipher)

- Symmetric key block cipher method.
- developed as various institute of standards & technology (NIST) in dec 2021

* General design of AES encryption cipher :-



Page No.
Date / /

*** The no. of round keys generated by the key expansion algorithm is always one more than the no. of rounds.

In other words, we have no. of round keys = Nr+1
we refer keys, $k_0, k_1, k_2 \dots k_n$

- Data units used in AES :-

There are 5 data units - bit, byte, word, block, state.

① Byte

Byte (b) → [b0 b1 b2 b3 b4 b5 b6 b7] (b) → $\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix}_b$

② word

word (w) → [B0 B1 B2 B3] → $\begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{bmatrix}_w$

③ block

| b0 | b1 | b2 | ——— | b15 |

④ State

$$S \longrightarrow \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

∴ each value of given matrix is 1 byte.

$16 \times 8 = 206$ bit.

each column represent word.

Page No.
Date / /

Que-1] for the given PT = "AES uses a matrix" convert into a state matrix.

→ As the given PT = 14 character
i.e 14×8 = 112 bit but we need to make it 128.
Now,
To insert 2 bogus character i.e z

Then it would be written as,
PT = 'AES uses a matrizz"

Convert character into the int values (Hexadecimal)

A	E	S	U	S	E	S	A	M	A	T	R	I	X	Z	Z
00	04	18	20	18	04	18	00	12	00	19	17	08	23	25	25

↓ int into hexadecimal

block = | 00 | 04 | 12 | 14 | 12 | 04 | 12 | 00 | 0C | 00 | 13 | 11 | 08 | 17 | 19 | 19 |

$$state = \begin{vmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{vmatrix} = \begin{vmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 17 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{vmatrix}$$

Que-2] PT = "THIS IS AN EXERCISE" convert into state matrix

→ convert given character into values:

T	H	I	S	I	S	A	N	E	X	E	R	C	I	S	E
19	07	8	18	8	18	0	13	04	23	04	17	02	08	18	04

Block: | 13 | 07 | 08 | 12 | 08 | 12 | 00 | 04 | 17 | 04 | 11 | 02 | 08 | 12 | 04 |

PT:
CT:

Teroz on 7.6

Page No.
Date / /

Round	Input state	out	Round key
Pre Round			

* AES round structure :-

Round = state matrix
- 4x4 matrix
- 16 byte
- column
- top to bottom.

* 4 types of transformation
- substitution → subtypes
- permutation → shift row
- mix input → mix columns
- Add round key ~~value~~ key values → Add round key

In these we will perform x-OR operation.

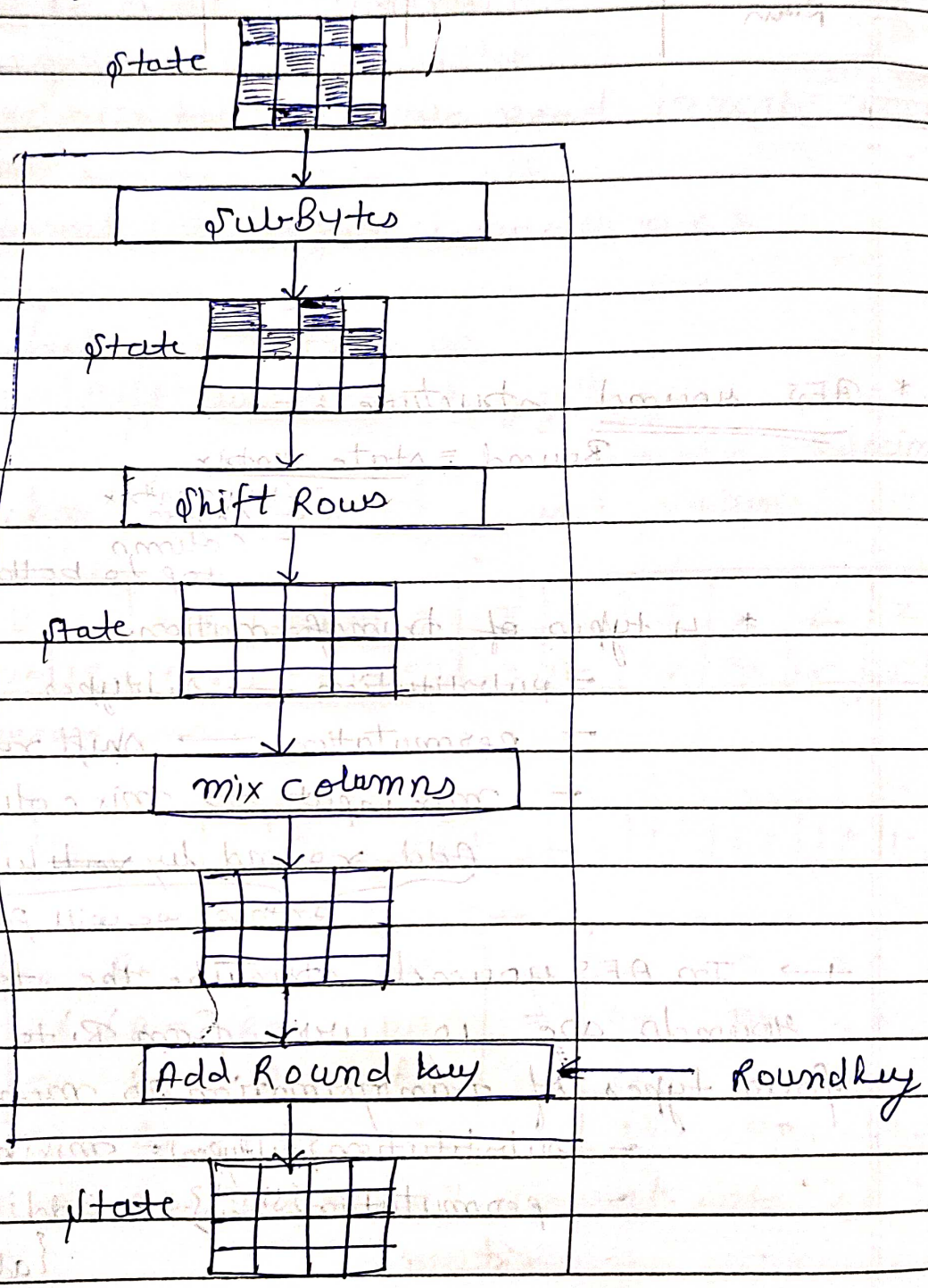
→ In AES round structure the total number of rounds are 10 were from 'R1' to 'R9', AES uses four types of transformation as mentioned below :-
- substitution {subtypes} - mixing {mix column meth}
- permutation {shifts} - adding round key values. {add round key}

- the input to each round is a state matrix and the output is given to each and every transformation as shown in figure below :-

- the last round i.e R10 perform the following transformation :- - substitution {subtypes}.
- permutation {shift row}
- adding round key values {add round key}

Page No.
Date / /

- following figure shows the above process:-



Page No.

Date / /

→ Substitution transformation → {SubByte method}

+ subbytes method (table form)

+ each byte is treated as two hexadecimal digits

*diagram :- + left digit = row value → 00 to FF(15)

+ right digit = column value → 00 to FF(15)

+ Intersection ⇒ (R, C) = new byte to be substituted.

+ Size of AES S-Box — 16 × 16

Q— convert the given state matrix using SubBytes transformation

State

00	12	0C	08
04	04	00	23
12	12	13	19
14	00	11	19

→

63	C9	FE	30
F2		63	26
C9		73	D4
FA		82	D4

→ Permutation transformation method → {shift row method}

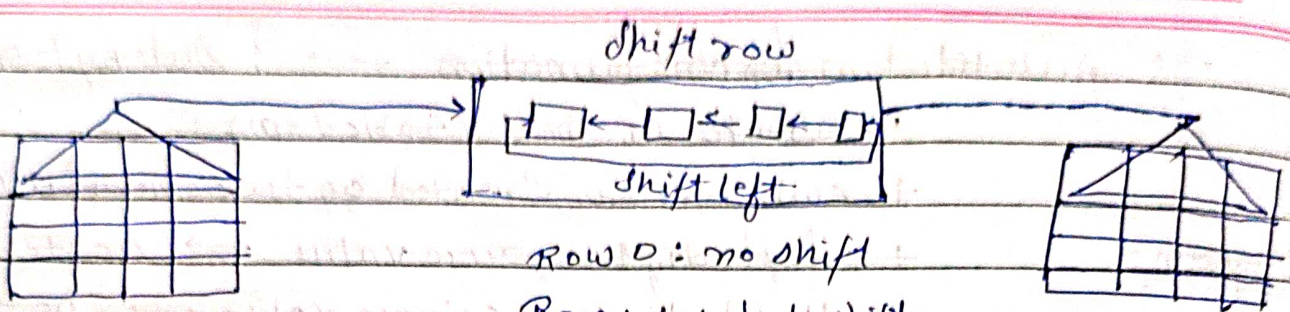
+ In these method permutation is done at the byte level.

+ shifting is done to the left

+ number of shifts depend on the row number value of the state matrix.

+ following figure shows the shift row method :-

Page No.
Date / /



Row 0 : no shift
Row 1 : 1-byte shift
Row 2 : 2-byte shift
Row 3 : 3-byte shift

Q- Perform shift rows transformation for the given state matrix

63	C9	FE	30
F2	F2	63	26
C9	C9	70	D4
FA	63	82	D4

State

→

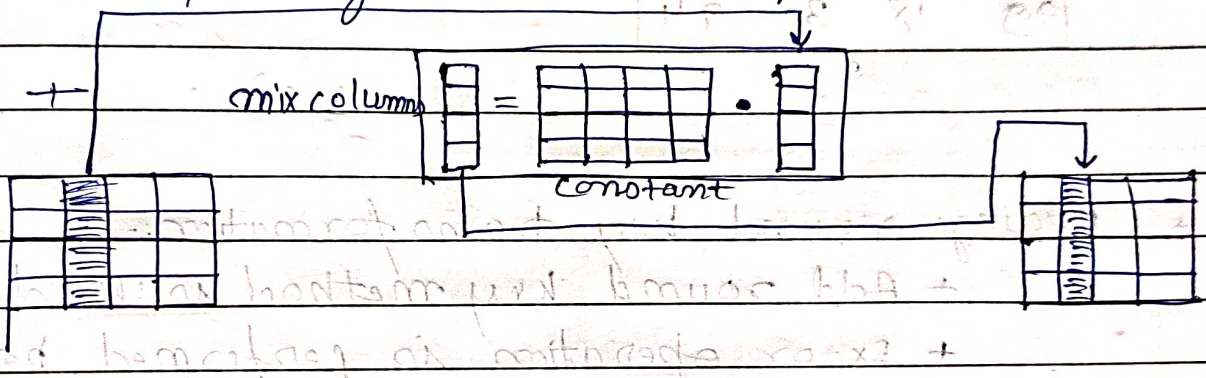
63	C9	FE	30
F2	63	26	F2
70	D4	C9	C9
D4	FA	63	82

{* write rules in paper if marks would be deducted}

Rounds	Input state	output state	round key

Page No.
Date / /

* Mixing Column transformation method:-
  + mix column method is operated at the column level.
  + matrix multiplication is done with a constant square matrix as shown below.
  + thus, these method combines or multiplies each byte with a different constant value and generates the output state matrix.
  + following shows the above process:-



  + constant matrix used by mix columns and Inv mix columns.

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

Inverse

0E	0B	0D	09
09	0E	0B	0D
0D	09	0E	0B
0B	0D	09	0E

Q- perfor mix column transformation for the given state matrix:-

⟶

63	C9	FE	30
F2	63	26	F2
7D	D4	C9	C9
D4	FA	63	82

Page No.
Date / /

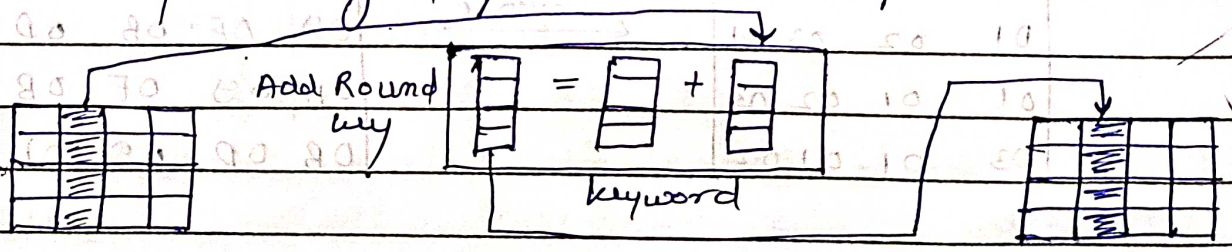
63	C9	FE	30		02	01	01	01
F2	63	26	F2	x	01	02	03	01
7D	D4	C9	C9		01	01	02	03
D4	FA	63	82		03	01	01	02

=

60	02	27	26
CF	92	91	0D
0C	0C	F4	D6
99	18	30	74

126 +

* Adding round key transformation:-
  + Add round key method is used.
  + Ex-or operation is performed between the input state matrix column with the given keyword values.
  + following figure shows the process:



Qu - perform add round key transformation for given matrices:

47	40	A3	4C		AC	19	28	57
37	D4	70	9F	⊕	77	FA	D1	5C
94	E4	3A	42		66	DC	29	00
ED	A5	A6	BC		F3	21	41	6A

Page No.

Date / /

→

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	82
1E	84	D7	D6

- A5 ⊕ 21

1010 0101 ⊕ 0010 0001

1000 0100 = 84

- 3A ⊕ 29

0011 1010 ⊕ 0010 1001

0001 0011

13

- 47 ⊕ AC

4×16¹ + 7×16⁰ ⊕ 10×16¹ + 12×16⁰

64 + 7 ⊕ 160 + 12

071 ⊕ 172

128 64 32 16 8 4 2 1

0 1 0 0 1 0 0 0

⊕ 1 1 1 0 1 0 1 1 0 0

1 1 1 0 0 1 0 0

- 47 ⊕ AC

0100 0111 ⊕ 1010 1100

128 64 32 16 8 4 2 1

1 1 1 0 1 0 1 1 = 219 = DB

EB

- E4 ⊕ DC

1110 0100 ⊕ 1101 1100

0011 1000 = 38

- 40 ⊕ 19

0100 0000 ⊕ 0001 1001

0101 1001

59

- A3 ⊕ 28

1010 0011 ⊕ 0010 1000

1000 1011

8B

- 4C ⊕ 57

0100 1100 ⊕ 0101 0111

0001 1011

1B

- 87 ⊕ 77

0011 0111 ⊕ 0111 0111

0100 0000

40

- D4 ⊕ FA

1101 0100 ⊕ 1111 1010

0010 1110

2E

- 70 ⊕ D1

0111 0000 ⊕ 1101 0001

1010 0001

A1

- 9F ⊕ 5C

1001 1111 ⊕ 0101 1100

1100 0011

C3

- 94 ⊕ 66

1001 0100 ⊕ 0110 0110

1111 0010

F2

- ED ⊕ F3

1110 1101 ⊕ 1111 0011

0001 1110 ⇒ 1E

Page No.

Date /

- 42 ⊕ 00

0100 0010 ⊕ 0000 0000

0100 0010

42

- A6 ⊕ 41

1010 0110 ⊕ 0100 0001

1110 0111

E7

- BC ⊕ 6A

1011 1100 ⊕ 0110 1010

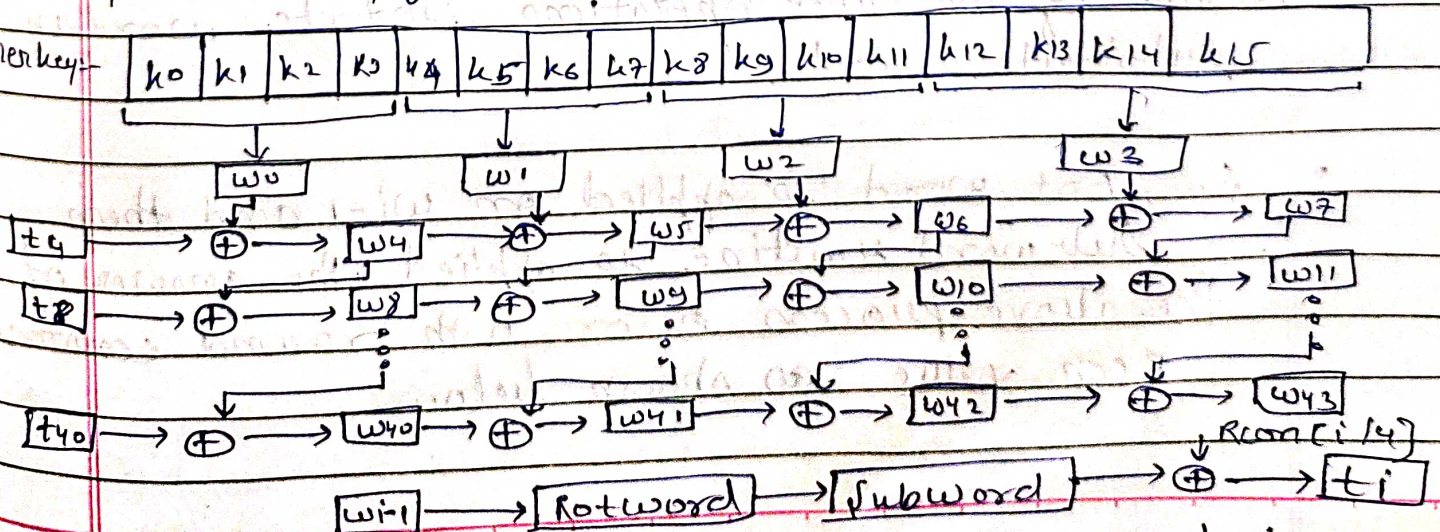
1101 0110

D6

Page No.
Date / /

Q-7 * AES key expansion process :- (128 bits)

- In AES key expansion process it creates (Nr+1) round keys of size 128 bit each
- from the given 128 bit cipherkey, it is used during preround transformation stage.
- Thus, the remaining round keys are generated from the given 128 bit cipher key
- Round keys are generated word by word were word = 4 bits thus, the expansion process creates $4 \times (Nr+1)$ words
- Thus, the word are called as ~~w0, w1, w2, ---- w4(Nr+1)~~
  $w_0, w_1, w_2, \ldots w_{4(Nr+1)-1}$ ~~(w0 - w43)~~
  128 bit — 44 words — R10 (w0 - w43)
  192 bit — 52 words — R12 (w0 - w51)
  256 bit — 60 words — R14 (w0 - w59)
- following figure shows the key expansion process for 128 AES :-



making of ti (temporary) words $i = 4Nr$

Page No.
Date / /

- from the above diagram the process is as follows:

- The given cipher key is of 16 bits i.e (128 bits) represents as k_0 to k_{15}

- The four words are created from the given cipher key where each word consist of 4bits.

- Therefore, $w_0 = k_0 - k_3$; $w_1 = k_4 - k_7$, $w_2 = k_8 - k_{11}$, $w_3 = k_{12} - k_{15}$

- The remaining words from w_4 to w_{43} are generated as follows: (wi for i=4 to 43)

(a) if $(i \bmod 4) \neq 0$, $w_i = w_{i-1} \oplus w_{i-4}$

(b) if $(i \bmod 4) = 0$, $w_i = t \oplus w_{i-4}$

here, t is a temporary word were,

t = Subword (Rot word (w_{i-1})) $\oplus$ [RCon i/4]

were, t is a temporary word that stores the result of the two operations rotate word and substitute word.

- So, Rot word is applied on w_{i-1} and then Sub word routine is applied the answer of the above process ex-or with round constants Rcon value as shown below.

Page No.
Date / /

Rcon Constants

Round	Constant (RKcon)	Round	Constant (RKcon)
1	(01 00 00 00)16	6	(20 00 00 00)16
2	(02 00 00 00)16	7	(40 00 00 00)16
3	(04 00 00 00)16	8	(80 00 00 00)16
4	(08 00 00 00)16	9	(1B 00 00 00)16
5	(10 00 00 00)16	10	(36 00 00 00)16

- (Rcon values) or the round constant values can be calculated using GF(28) and using irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$).
  {(refer pg. 236) of forzon E-book for process}

- following shows the pseudocode for 128 bit AES:-

```
KeyExpansion ([key0 to key15], [w0 to w43])
{   for(i=0 to 3)
        wi ← key4i + key4i+1 + key4i+2 + key4i+3
    for(i=4 to 43)
    {
        if (i mod 4 ≠ 0)  wi ← wi-1 + wi-4
        else
        { t ← SubWord(RotWord(wi-1)) ⊕ Rcon[i/4]
          wi ← t ⊕ wi-4
                // t is a temporary word
        }
    }
```

Page No.
Date / /

Q – Alice and Bob agreed upon a 128 bit cipher bit as given as :-

k0	k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
24	75	A2	B3	34	75	56	88	31	E2	12	00	13	AA	54	87

Calculate the keys for each round for AES 128 bit version.

→

round	value of t	first word in the round	second word in the round	Third word in round	fourth word in the round
Pre-round (K0)	–	w0 = 24, 75, A2, B3	w1 = 34, 75, 56, 88	w2 = 31, E2, 12, 00	w3 = 13, AA, 54, 87
R1	AD20177D	w04 = 8955B5CE	w05 = BD20E346	w06 = 8CC2F1[illegible]	w07 = 9F66[illegible]
R2		w08 =	w09 =	w10 =	w11 =

Round 1

– w04 = In pseudocode.

i=4 if (4 mod 4 ≠ 0) // false

else :–

t ← subword (Rotword (wi-1)) ⊕ Rcon i/4

– Rotword (13AA5487)

~~subword (13AA5487) = AA548713~~

~~AA548713 ⊕ Rcon.~~

subword (AA548213) ⊕ Rcon i/4

– AC20177d ⊕ Rcon1

– AC20177d ⊕ 01 00 00 00

Page No.
Date / /

- AC2017 7D ⊕ 01 00 00 00

1010 1100 0010 0000 0001 0111
0000 0001 0000 0000 0000 0000
1010 1101 0010 0000 0001 0111

AD2017 7D

0111 1100
0000 0000
0111 1100

t ← AD2017 7D

W4 ← t + W0
W4 ← AD2017 7D ⊕ 75A2B3
W4 ← 8955B5CE

- W05 = i=5
W5 ← W4 ⊕ W1
8955B5CF ⊕ 34755688
W5 ← BD20E346

- W06 = i=6
W6 ← W5 ⊕ W2
W6 ← BD20E346 ⊕ 31E21200
W6 ← 8CC2F146

- W07 = i=7
W7 ← W6 ⊕ W3
W7 ← 8CC2F146 ⊕ 13AA5487
W7 ← 9F68A5C1

* Round 2
- W08 = In pseudocode.
i=8 if (4 mod 4 ≠ 0) - false

Page No.

Date / /

$t \leftarrow$ subword(Rotword(w_{i-1})) $\oplus$ Rcon$_{i/4}$

$t \leftarrow$ subword(Rotword(9F 66 A5 C1)) $\oplus$ Rcon$_2$

$t \leftarrow$ subword(66 A5 C1 9F) $\oplus$ Rcon$_2$

$t \leftarrow$ subword(33 06 78 DB) $\oplus$ Rcon$_2$

$t \leftarrow$ 33 06 78 DB $\oplus$ 02 00 00 00

$t \leftarrow$