UNIT-II

1) BLOCK CIPHER PRINCIPLES AND ALGORITHMS :-

Block Cipher :-

↳ Our main moto is to convert PT into CT. So, the third person cannot understand what we are sending.

↳ We are dividing our plain text, is divided into no. of blocks.

↳ Suppose we have abcdef as a plain Text. We will be dividing it into no. of blocks.

2 Blocks

↳ After dividing the plain text into no. of blocks we will be converting each individual block into a cipher Text block.

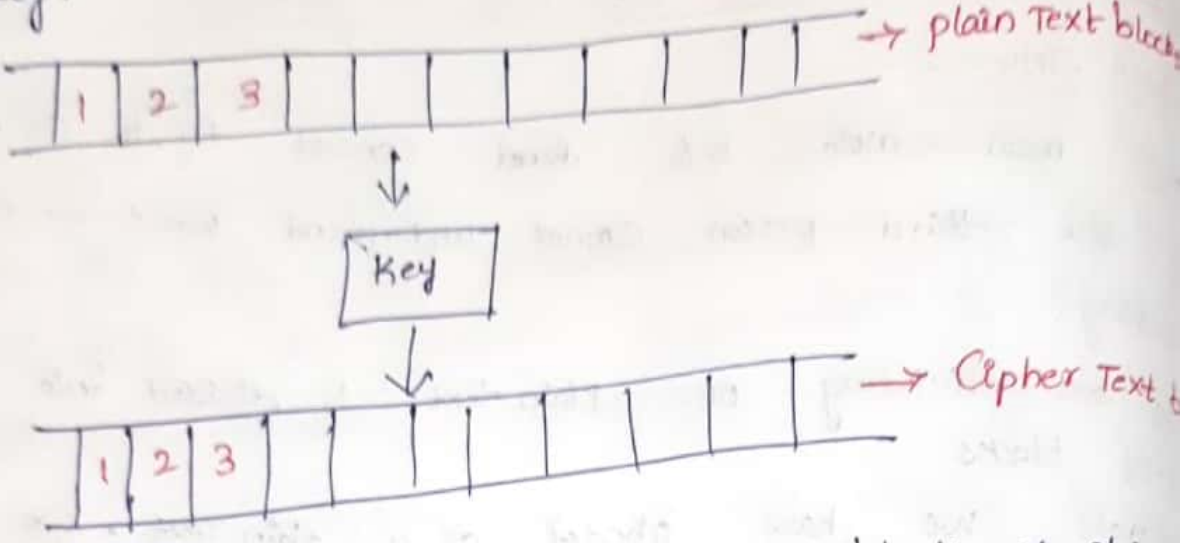
Block size (40, 56, 64, 128, 256 bits)

Whenever, we are dividing the blocks then, we are converting the PT to CT. Always, make sure that PT block size should be equal to CT block size.

PT block size = CT block size

Ex:- Block size of PT = 40 bit
The corresponding CT = 40 bit
} It should be in same size.

↳ The conversion of PT to CT is happen through Key.



↳ For each and every single block of plain the individual CT block is generated.

↳ Whatever PT you have you will be dividing no. of blocks. After dividing PT into ^ blocks you will be using Key, with that Key you will convert into CT.

↳ After generating the corresponding CT ~~blocks~~ now we have to combine all CT blocks.

abcdef

PT [ab] [cd] [ef] For each & every text by using a Key we generate CT.

↓K ↓K ↓K

CT [xy] [kl] [mo]

xykImo

This xykImo is corresponding CT for abcdef corresponding PT.

We have several algorithms to generate CT. After generating CT, this CT will send to

receiver side, on reaching receiver side again receiver use the key and he will generate corresponding PT. Then he will combine all the blocks & he will read actual msg.

Block Ciphes principles:-

Three design principles are there. They are:

1. Numbes of Rounds → 10R, 16R, 20R.
   - ↳ Each & every algorithm has several Rounds.
   - ↳ How many higher no. of rounds will have that much tough the algorithm to the third party person to break it.
   - ↳ How many rounds will be more that much hard will become to the hacker.

   No. of Rounds should be more.

2. Design of function F:- We will have a function based on that function only means like $f(x) = ax+b$
   - ↳ You must design a function 'f' which will be very much complicated to understand. How much harder the function is that much more time the hackers takes to Decode or to break the algorithm.

   We should take **Non-linear functions**. bcz they are complicated.

3. Key Schedule Algorithm: You should be careful while generating a Key because Key is very important. If we are having minor change in Key lot of

changes will be there.

Block ciphers modes of Operations:-

ECB - Electronic Code Book
CBC - Ciphes Block chaining
CFB → Cipher Feeed Back mode.
OFB → Output Feed Back mode
CTR → Counter modes.

Block Ciphes Algorithms:-

1. Data Encryption Standard
2. Advanced Encryption Standard
3. Blowfish Algorithm

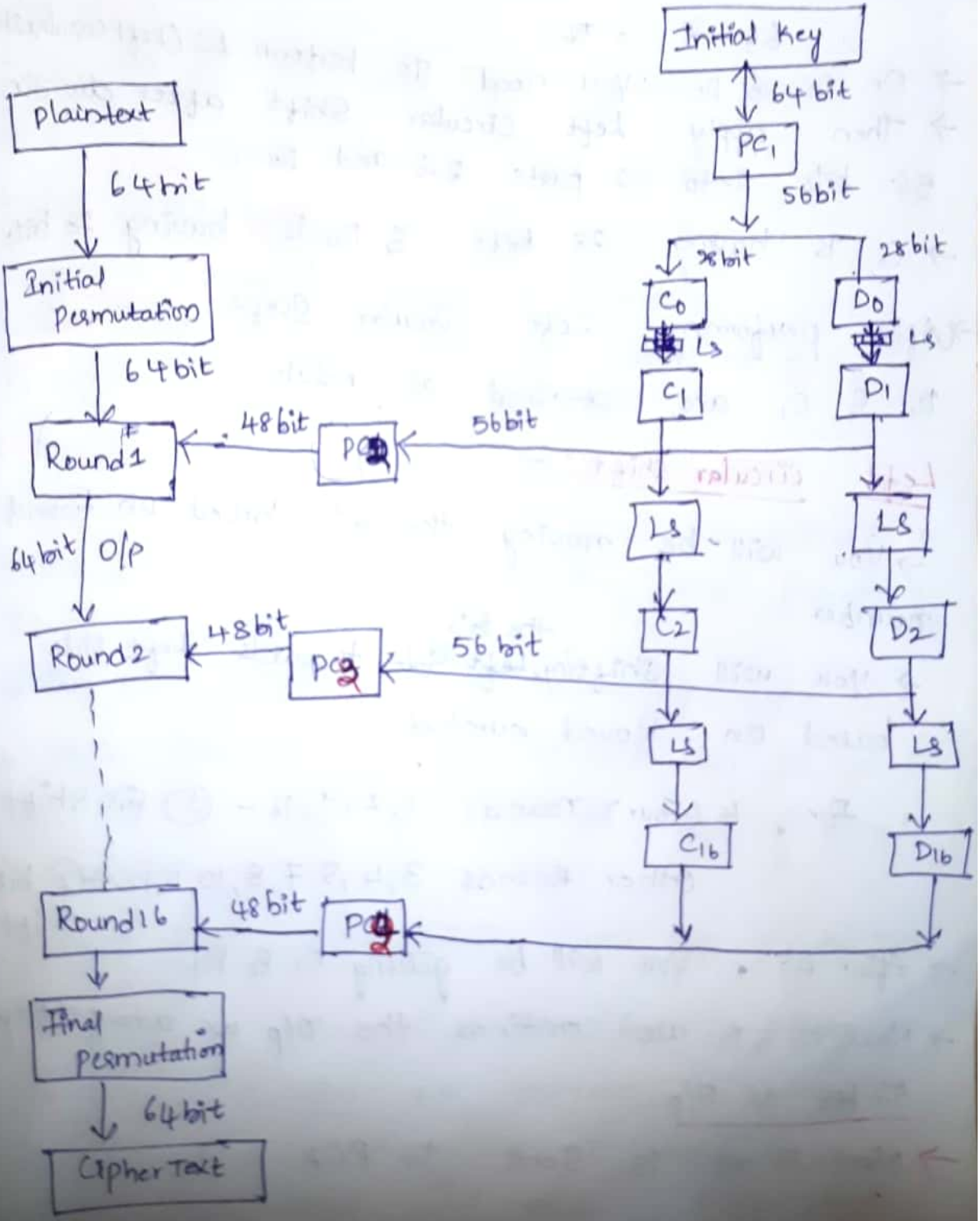
DES Algorithm:- (Data Encryption standard)

↳ Converting PT to CT.
↳ It is Block Cipher Algorithm.
It has total of 16 Rounds.
Text size = 64 bits → PT.
Key size = 48 bits.
Why we got 48 bits as Key Size means 16 bits are gone. 8 bits are removed for parity and 8 bits for rearrangement.
In each round 4 steps are performed. Total 16 Rounds will be there. In each & every round 4 steps will be there inside.

4 steps:

1. Dividing Bits into 2 parts as 32-bit each.
2. Bit shuffling.
3. Non linear Substitutions.
4. Exclusive OR Operations.

DES Algorithm



→ First we have Initial key which is of 64 bit
→ 64 bit will be sent as input for PC1.

Inside PC1 (Permuted choice 1)

→ From 64 bits, 8 parity bits are to be removed from every 8th position.

In 64 → 8×8

∴ 64 − 8 = 56.

→ On C_0 & D_0 you need to perform LS (Left Circula
→ Then apply Left Circular Shift after divi
56 bits into 2 parts : C_0 and D_0.

→ C_0 is having 28 bits & D_0 is having 28 bi

→ After performing Left Circular Shift
D_1 & C_1 are obtained as result.

Left circular shift :-

→ You will be moving the bits based on Round number.

→ You will Shifting the bits towards Left Side based on Round number.

For, 4 (Four) rounds 1, 2, 9, 16 – (1) bit shift
Other Rounds 3, 4, 5, 7, 8, 10, 11, 12 → (2) bit Shift

→ After LS, you will be getting C_1 & D_1.
→ Now, C_1 & D_1 are combined the o/p we are getting 56 bit as o/p.
→ Now, 56 bit is sent to PC2.

In PC2:-

↳ C_1 and D_1 are combined to form 56 bits again. Permuted Choice 2 is applied.

↳ 56 bits are rearranged, they are permuted and 48 bits are selected.

↳ We remove 8 bits from 56 bits then it became 48 bits.

↳ This 48 bits will send for Key for round 1

→ I/p for round 1 is 48 bits.

→ This 48 bit Key i/p is given for Round 1.

→ Round 1 o/p is 64 bit. is generated.

→ How 48 bit Key is generated, on C_1 & D_1 LS is applied then C_2 & D_2 will get.

→ Combine C_2 & D_2 then it give to PC2

→ 56 bit is given to PC2 will do permutat^n, rearrangement & select^n & generate 48 bit Key for Round 2.

→ This same process will be repeating upto 16

→ After Round 16 you get 64 bit Key this you will be applying final permutation

→ After final permutation you get Cipher Text.

↳ This is the Algorithm to convert PT to CT. In this we perform 16 rounds.

↳ for each & every round we have 4 Substeps in that. and each & every round

what is the i/p we are giving the o/p of previous round plus 48 bit key.

→ For all the rounds the 48 bit key will not given. It will have different key for diff rounds.

→ Because, inorder to secure the algorithm → the third party get the key corrupt the algorithm. So, we use different keys for diff rounds.

BLOCK CIPHER MODES OF OPERATION!

Dividing PT into diff no. of blocks is Block Cipher.

And when we are doing operation on that block cipher we have diff modes. Totally we have 5 modes

1) Electronic Code Book (ECB):-

1. First we will do PT into no. of blocks
2. We will be encrypting the PT with the help of a key.
3. In order to Encrypt we need PT & Key
4. In order to decrypt we need CT & Key
5. PT & key is given as input

DES:- (Data Encryption Standard)

The DES is a symmetric key algorithm for the encryption of digital data.

Developed in the early 1970's by IBM and was published as an official Federal Information Processing standard in 1976.

→ In our algorithm we take 64 bit key.

Key = 0001001100110100 0101011101111001 1011100

This is 64 bit key. 1101111111110001

↳ The 64-bit key is permuted accordingly to the following table, PC-1.

↳ In this as per rule only 56 bits of the original key appear in the permuted key.

PC-1 Table

57	49	41	33	25	17	9
1	58	150	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	21	4

as per table we should arrange the key values.

↳ 57th bit in key will be our 1st element.

56 bit key = 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001110

→ In table 7 rows and 8 columns are there.

→ 7×8 = 56 bit,

The no. of bits in o/p that is equal to the no. of digits in the pc1 table.

→ After getting 56 bit key then Next, Split key into left and right halves, C_0 and D_0, where each half has 28 bits.

from the permuted key, we get.

C_0 = 1111000 0110011 0010101 0101111

D_0 = 0101010 1011001 1001111 0001110

Iteration Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Number of Left shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

By applying this Left shift on C_0, D_0 it becomes C_1 & D_1 as the Left shift will be done.

C_1 = 1110000 1100110 0101010 1111

D_1 = 0101010 1100 1100 1111 0001 1110

→ After getting upto C_{16} to D_{16}.

Next, Combining of C_1 & D_1 we get again 56 bit. That 56 bit o/p is given to PC-2 table.

After taking pc-8 table then according to pc-2 table only 48 bit output will be come.

→ Each pair has 56 bits, but PC-2 only uses 48 of these.

C_1 = 11100001100110010101011111

D_1 = 10101010110011001111000111110

Together, Combinely we get 56 bit as.

C_1D_1 = 11100001100110010101010111111 10101010110011 00111100011110

PC-2 table gives 48 bits

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

8 x 6 = 48

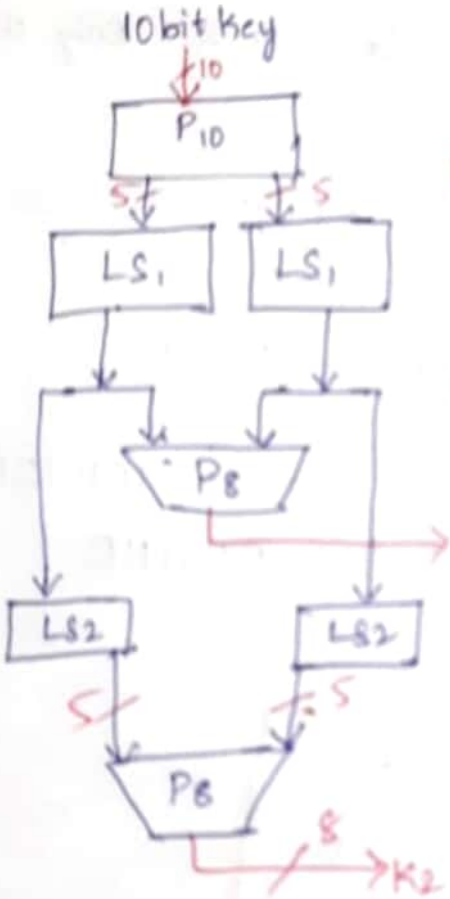
After applying pc2 on C_1, D_1 we get o/p

as.

K_1 = 0001101100 00101011 10111100 01100000

SDES:-

Key Generation:-



IP →

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

P_{10}

1	2	3	4	5	6	7	8	9	
3	5	2	7	4	10	1	9	8	

P_8

1	2	3	4	5	6	7	
6	3	7	4	8	5	10	

IP^{-1}

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

E/P

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

P_4

1	2	3	4
2	4	3	1

In binary forms.

$$S_0 = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{bmatrix}$$

$$S_1 = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{bmatrix}$$

Key Explanation:-

Key

1010 010 0010 → 10 bit Key

01000 00100 → we divide 5 bits & 5 bits. we will apply One key numbers to Left Shift.

K_1 → 00001000 → After applying P8 '10' becomes key will become '8' bits. This is Key 'K_1' = 00001000

After generating Key 'K_1' we will do Left Shift with two numbers. Again we will take 10 bit key and do Left shift again with 2 numbers

00000 10000

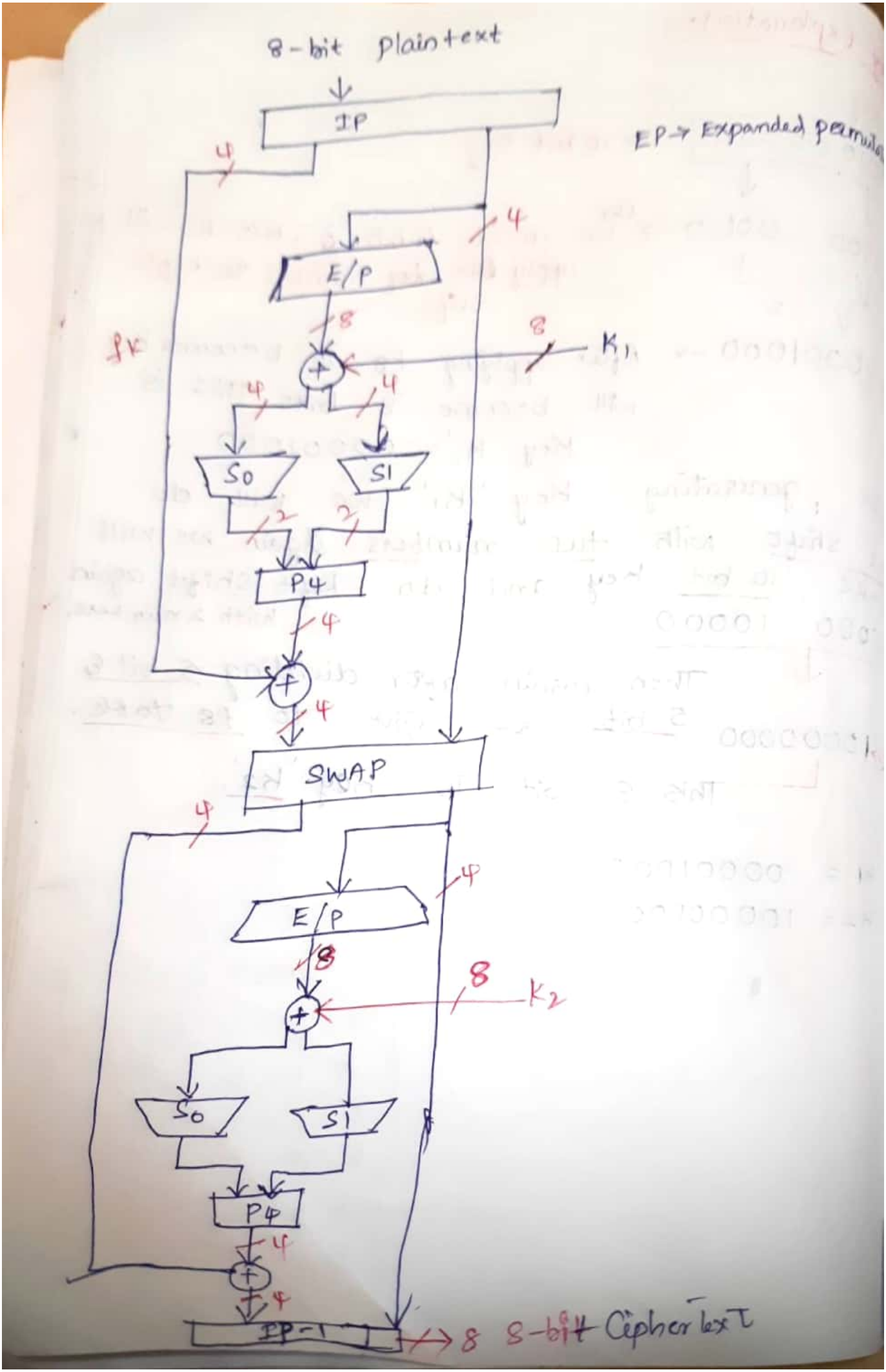
Then again after dividing 5 bit & 5 bit we give to P8 table.

K2 → 10000000

This 8 bit is Key 'K_2'.

K_1 = 00001000

K_2 = 10000000



8-bit plain text = 11001000. We divide into 4 bits, 4 bits.

This 8 bit gives to IP table. After giving 8 bit to IP table.

L R
10010010 We will take right 4 bits and we will give to Ep table.

0010 will be applied → E/p table. After giving right 4 bits to E/p table then we get,

00010100 → E/p table after giving 4 bits to E/p then
XOR ⊕ 00001000 — K1 we take K1 to do XOR operation

00011100
S0 S1

S0 → 0001 → 1st & Last bit is for row.
S1 → 1100

2^2 2^1 2^0
4 2 1
0 1 → 1

S0 → row → 01 = 1st row } 3 = 11
S0 → Column → 00 = 0th column } After doing decimal to binary Conversion we get.

4 2 1
0 0 → 0

S1 → row → 10 = 2 } 1 → 01
S1 → Column → 10 = 2 }

4 | 2 | 1
1 0

2 | 3 - 1
1 -
1 | 1 - 0
r

1101 → This 4 bits we will give to P4 table

1101 → After giving to P4 table then is from left part

1101
XOR ⊕ 1001 → This is 4 bits after giving to IP table.

01000010 — These are right bits of after IP.

then we should swap right to left, left to right the bits will be swapped.

R
00100100 → we should give this to E/p table.

After giving to E/P table.
Only right bits will give

00101000 → After

XOR ⊕ 10000000 → RK

00101000 ⊕ 10000000

01010100

S0 S1

10101000

S0 S1

S_0 = 1010

S_1 ⇒ 1000

3 2 1 0
2 2 2 2
8 4 2 1
1 1

8 4 2 1
0 0
1 0
2 2

S_0 = row = 10 ⇒ 2 (row) } 2 = 10

S_0 = Column ⇒ 01 ⇒ 1 (col)

S_1 = row = 10 ⇒ 2 (row) } 3 = 11

S_1 = Column = 00 ⇒ 0 (col)

1011 → This will give to P4 table

0111 ⊕ 0010

0101 0100

L R

These are ri
bits from
after Swapp

After we should do IP⁻¹

This is 10000101 the 8-bit Ciphertex

CT = 10000101

ADVANCED ENCRYPTION STANDARD (AES)

Plain Text = 128 bits size.

Cipher Text = 128 bits size.

→ Here Key size is be 128, 192, 256 bits

No. of rounds	Key length (Bytes)
10	16
12	24
14	32

If we are having 128 bits → 10 rounds will be there

192 bits ⇒ 12 rounds will be there

256 bits ⇒ 14 rounds will be there.

→ plaintext 128 bit is represented as 4×4 column matrix.

→ To perform encryption & decryption operation the 128 bits are represented in 4×4 column matrix.

↓

This is called State

→ The matrix is called as State.

→ Each block size is 16 bytes.

Block diagram :-

First PT 128 bits are converted to 4×4 matrix is called State.

→ 128 bits are stored in State array.

→ Next input state is given to Initial transformation. Where this

→ Initial transformation is going to do some Operation with this Input State Array values then that transformed value is going to given to Round 1.

→ Where in Round 1 we are having 4 transformations

→ We Round 1, Round 2, Round N-1, & R
N. Here, why it is mentioned as
instead of Numbers. We will see later

The 4 transformations are :-

1. Substitute bytes
2. Shift rows.
3. Mix Columns.
4. Add round Keys.

→ In Round '2' also 4 transformations are going to happen. When the output is given round 3, round 4 so on upto round N-1, round N-1 having 4 transformations & in only 3 transformations.

→ Incase we are having 12 rounds then Roun to Round 11 we are having 4 transform & in Round 12 there are 3 transformations

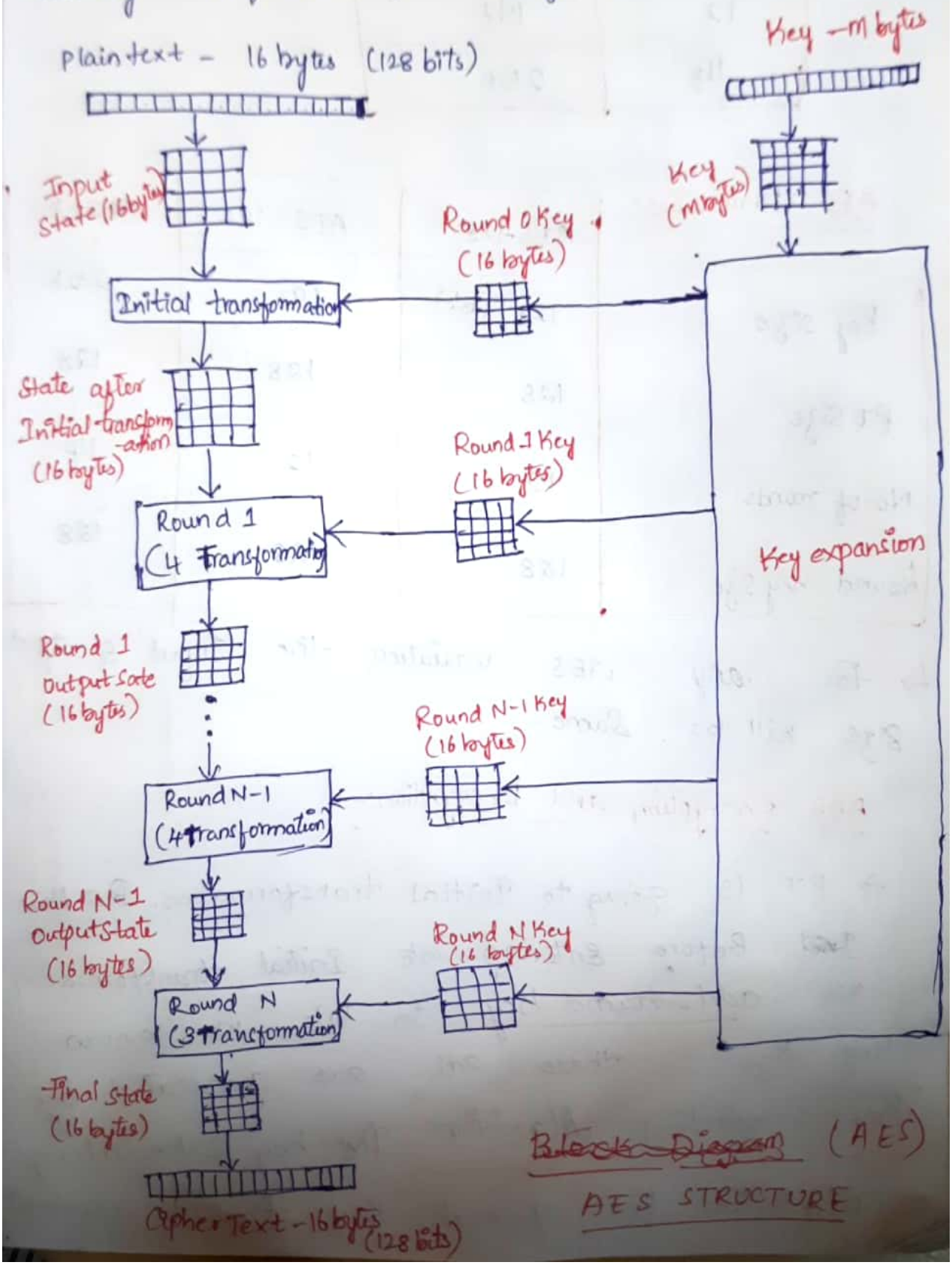
→ Whatever we are getting in Round N th is finally stored in State that 16 bytes C

→ We note here, for every round we need a roundKey. for processing.

→ All these round keys are 16 bytes.
↓
means = 128 bits

The original key size is 'M' bytes which

means there is some relationship between the Key Size and number of rounds performed in AES. Key is used by Key Scheduling Algorithm. Inorder generate a SubKeys or round Keys that is actually required for every round.



~~Block Diagram~~ (AES)

AES STRUCTURE

The Relationship between No. of rounds and

No. of rounds	Key size (in bits)
10	128
12	192
14	256.

AES Parameters:-

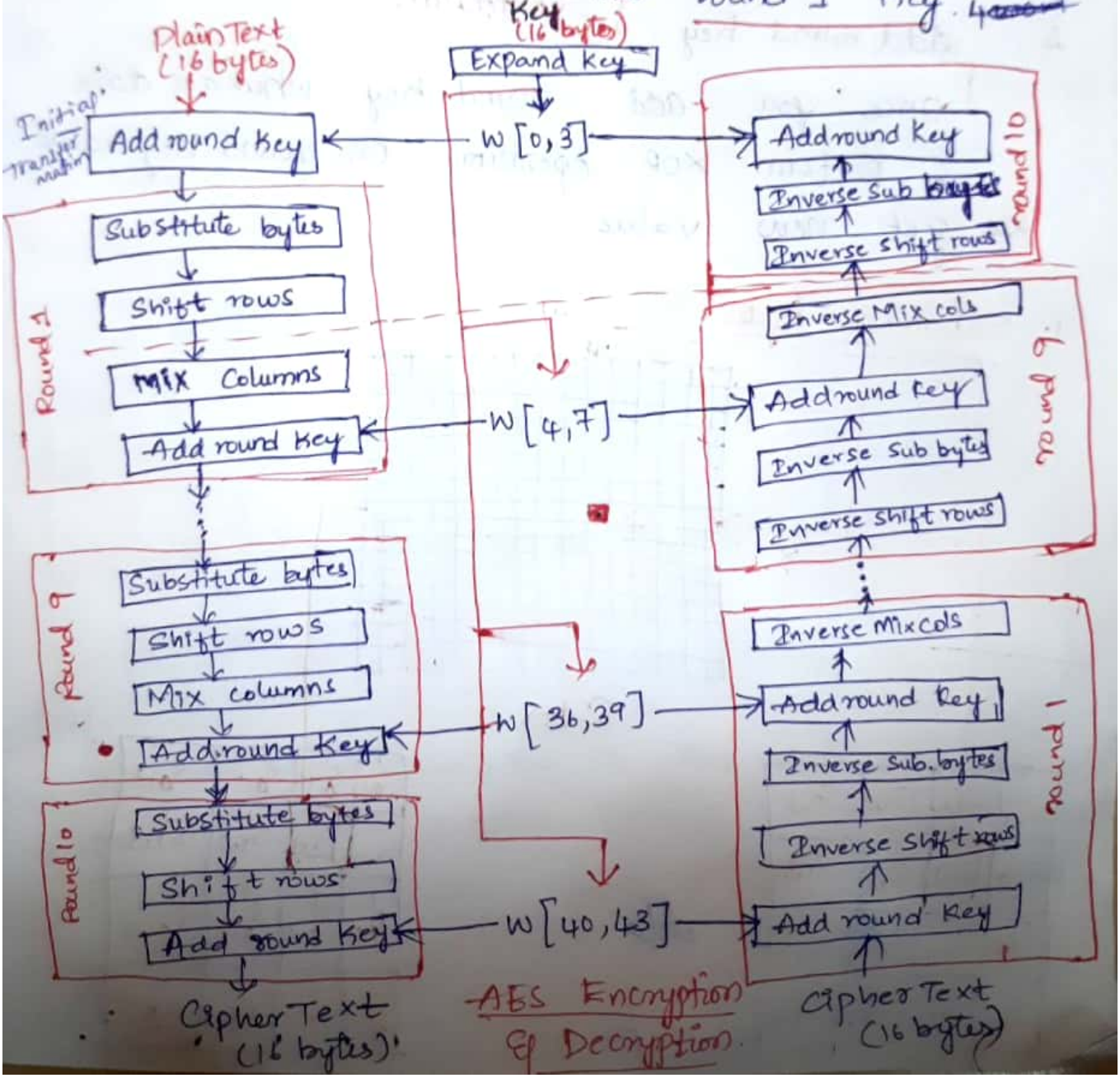
	AES-128	AES-192	AES-2
Key size	128 (bits)	192	25
PT size	128	128	128
No. of rounds	10.	12	14
Round Key size	128	128	128

↳ For any AES variation the Input & outp size will be same.

AES Encryption and Decryption:-

→ PT is going to Initial transformation. But th Before entering into Initial transforma we add round key. In this we know Key size. these all are generated by "Key scheduling algorithm". The key schedulin algorithm first generates key that is requ

for the Initial Transformation that is adding round Key to the PT. The 1 word W[0,3] is generated by Key Scheduling Algorithm. the words makes with 32 bits. 0,1,2,3, 4 words 4×32 bit = 128 bit that is generated by Key scheduling algorithm. for the Initial Transformation. The plain text 16 bytes (128 bits) is added with 128 bit Key this Key is the round Key for the Initial Transformation which is refered as round '0' Key. Once XOR ⊕ operation is Performed then it enter into round '1' Key.



AES Encryption & Decryption

(10 rounds)
→ 44 words are required and Expanded by key schedu algo
→ 12 rounds 52 are required for AES-192
→ 14 rounds 60 are required for AES-256.

→ Here, we are taking Key Size as 128 bits (16b
and perform Key scheduling Algorithm.

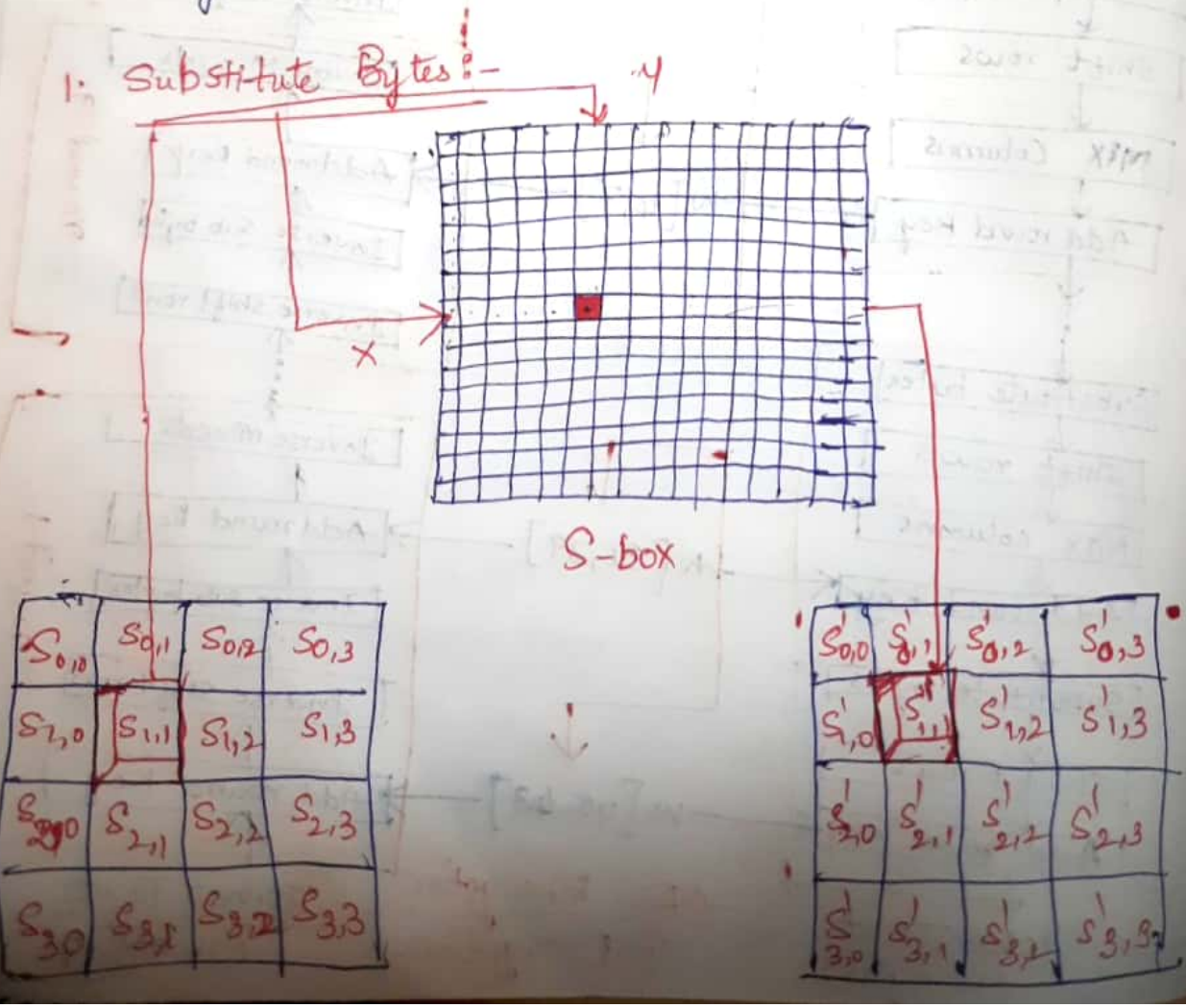
AES Round Transformation :- The key size are basic word Key Size

PT is converted into CT.

1. Substitute Bytes → This is Substitution Techniq
2. Shift Rows → this is Permutation
3. Mix columns } Substitution
4. Add round Key.

Once you add round key whenhave data & perform XOR operation on round key va we get new value.

1. Substitute Bytes :-



→ This S-Box is having 16 × 16 matrix.

→ For Every individual cell is going to be replaced by other values.

→ For Ex: $S_{0,0}$ is replaced with $S'_{0,0}$.

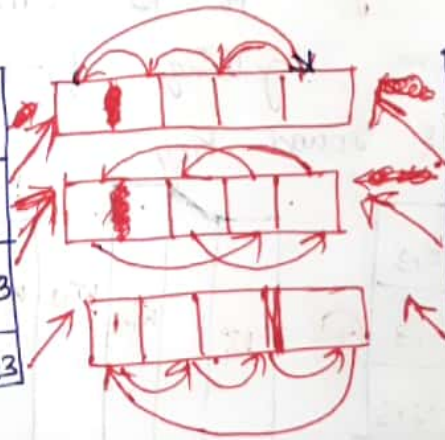
→ All values are Substituted with other new values by looking up this Substitute box (s-box) table.

→ Lets assume this value ($S_{1,1}$) is exactly matching to this 'x' position & 'y' position. So, whatever is there in the intersection value that value is replaced as $S'_{1,1}$.

→ Sub bytes is simply a lookup table. One value is replaced with other values as per as table.

2. Shift Rows :-

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$



$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,1}$	$S_{1,2}$	$S_{1,3}$	$S_{1,0}$
$S_{2,2}$	$S_{2,3}$	$S_{2,0}$	$S_{2,1}$
$S_{3,3}$	$S_{3,0}$	$S_{3,1}$	$S_{3,2}$

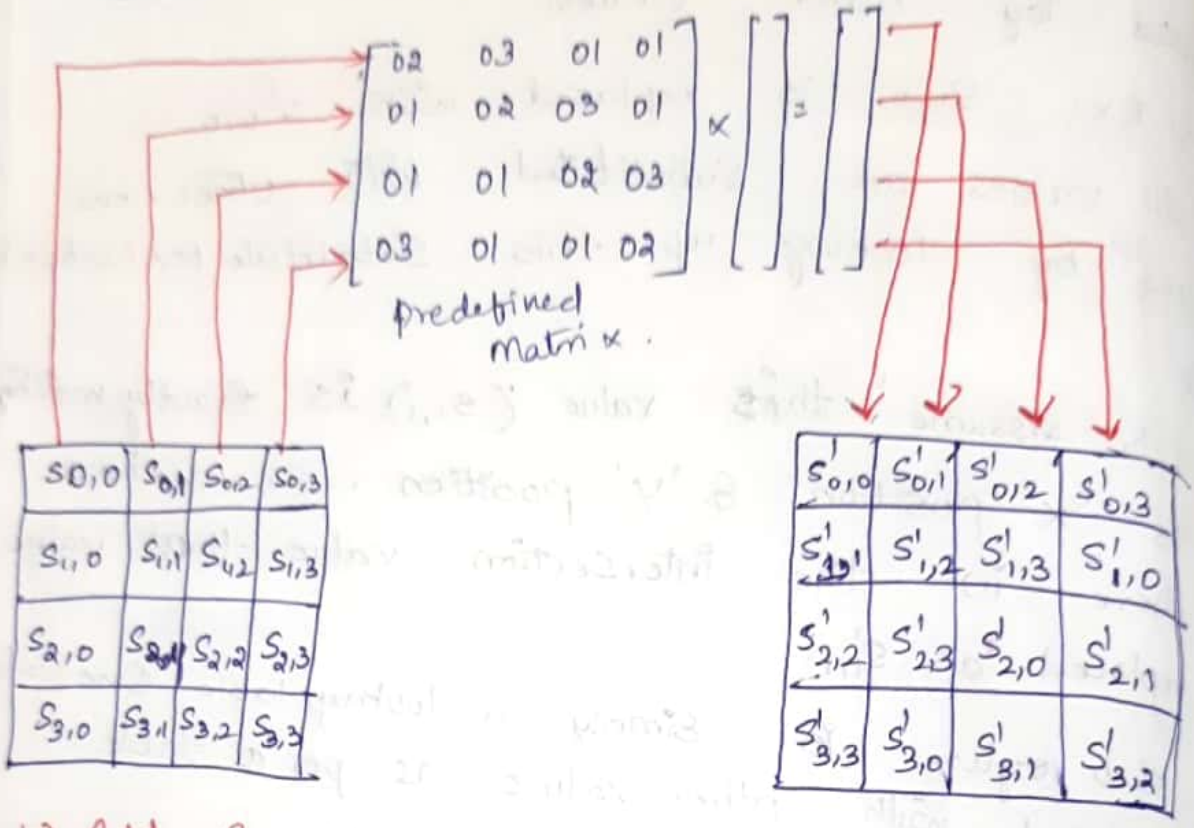
→ In 1st row there is no shifting operation is performed.

→ In 2nd row Shifted to One - one place (one byte Left circular shift)

→ In 3rd row shifted to two - two place. (two byte LCS)

→ In 4th row one moving. (3 byte LCS)

3) Mix Columns :- Matrix Multiplication is performed Order is 4 × 4 matrix.



$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$s'_{0,0}$	$s'_{0,1}$	$s'_{0,2}$	$s'_{0,3}$
$s'_{1,0}$	$s'_{1,1}$	$s'_{1,2}$	$s'_{1,3}$
$s'_{2,0}$	$s'_{2,1}$	$s'_{2,2}$	$s'_{2,3}$
$s'_{3,0}$	$s'_{3,1}$	$s'_{3,2}$	$s'_{3,3}$

4) Add Round Key :- It is simply XOR (⊕) operation Bitwise.

4 words are getting from key scheduling algorithm as round key.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

⊕

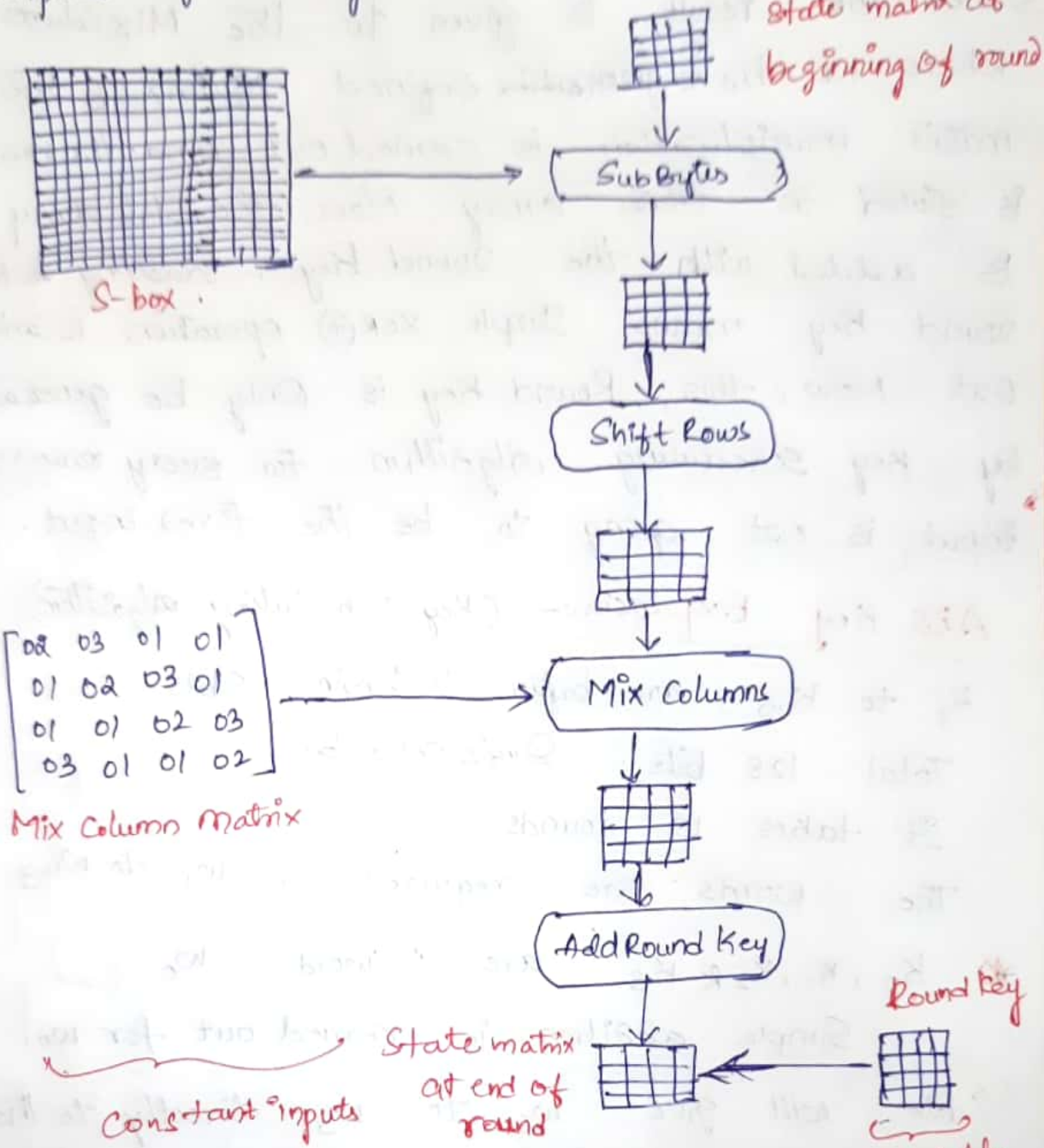
=

$s'_{0,0}$	$s'_{0,1}$	$s'_{0,2}$	$s'_{0,3}$
$s'_{1,0}$	$s'_{1,1}$	$s'_{1,2}$	$s'_{1,3}$
$s'_{2,0}$	$s'_{2,1}$	$s'_{2,2}$	$s'_{2,3}$
$s'_{3,0}$	$s'_{3,1}$	$s'_{3,2}$	$s'_{3,3}$

These are given to round Keys.

In single Round

→ Inputs for Single AES Round.



→ Input state is ~~taken~~ ~~from~~ given to Sub bytes by look up table is performed on Input state.

After substitution technique it is going to shift rows.

↳ Where the rows are shifted accordingly, the

↳ First row is untouched.

↳ Second row is performed with 1 byte LCS. (Left circular Shift)

↳ third row is performed with 2 bytes LCS

↳ fourth row is performed with 3 bytes LCS

→ Then the result is given to the Mix column where we have predefined Matrix & it matrix multiplication is carried out and the result is stored in State array. Now, the State Array is added with the round key. Adding round key means simple XOR (⊕) operation is carried out. Now, this, Round Key is only be generated by Key scheduling Algorithm. for every round input is not going to be the fixed input.

AES Key Expansion:- (Key scheduling Algorithm)

K_0 to K_{15} are bytes. ⇒ 1 byte = 8 bits

Total 128 bits. Single cell 8 bits.

It takes 10 rounds.

The words are required are w_0 to w_{43}

* K_0, K_1, K_2 & K_3 are 1 word = w_0.

Simple addition is carried out for w_0.

↳ We will give w_0 to w_3 directly to the Initial Transformation. We need 44 words by using w_0, w_3 from these 4 words the 44 words are generated. that is key using the Key Expansion process.

↳ From the original key we are generated 4 words. which is sent the initial Transformation as round Key. These w_0 to w_3

these 4 words are going to be use for generating next 4 words.

→ w_3 is given to 'g' function xor ⊕ of w_0 ⇒ w_4

→ w_4 o/p is given to w_1 & perform xor ⊕ to get = w_5

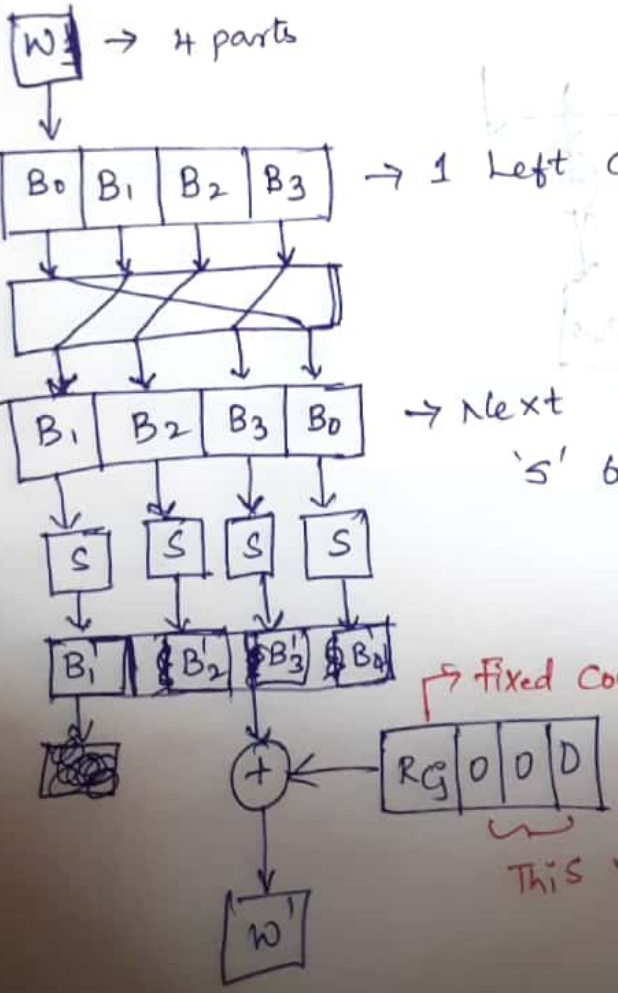
→ w_5 o/p & w_2 i/p these two will perform ⊕ to get = w_6.

→ w_6 o/p & w_3 i/p will perform ⊕ to get w_7.

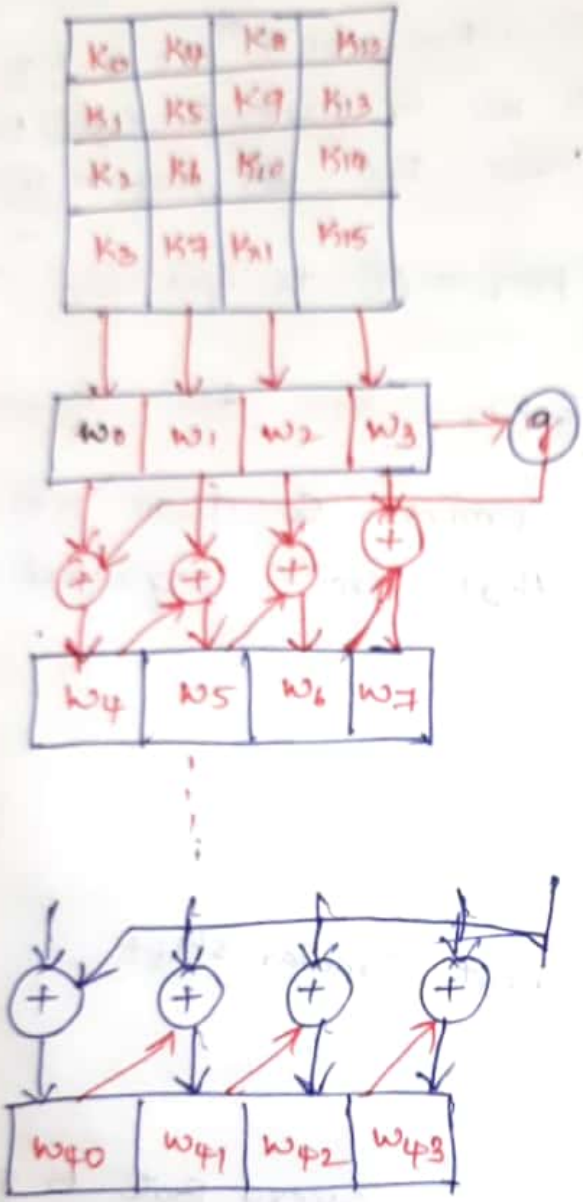
These w_4, w_5, w_6 & w_7 are for Round 1.

→ Next, repeat this process so, that we will get all Round Keys that are required for the AES operation.

'g' function:-



AES Key Expansion :-



Blowfish :-

↳ Blowfish is a symmetric-key block cipher, designed in 1993 by Bruce schneier and included in many ciphe and encryption products. Blowfish provides a good encryption.

↳ This is replacement of DES & IDEA Algorithm.

↳ The size of PT is 64 bits

↳ The size of key varies from 32 to 448 bits. Default is 128 bit.

↳ No. of rounds are 16.

→ In Blowfish how the key is Expanded :-

→ Our Original key is divided into 18 subkeys

→ All subkeys are $(P_1, P_2 \cdots P_{18})$ P-arrays.

→ Along with the division of keys into different no. of sub keys. Our key is also divided into different no. of S-boxes.

→ We are deriving 32-bit S-box from original key.

→ Each S-box contain approx 256 entries.

→ If we want P_1 → Initialize P-array, sbox with null value.

$P_1 \rightarrow P_1$ XOR K_1 (First 32 bits of keys)

$P_2 \rightarrow P_2$ XOR K_2 (with second 32 bits of keys)

P_{14} XOR K_{14}

P_{15} XOR K_{15} (with Eighteenth 32 bits of keys)

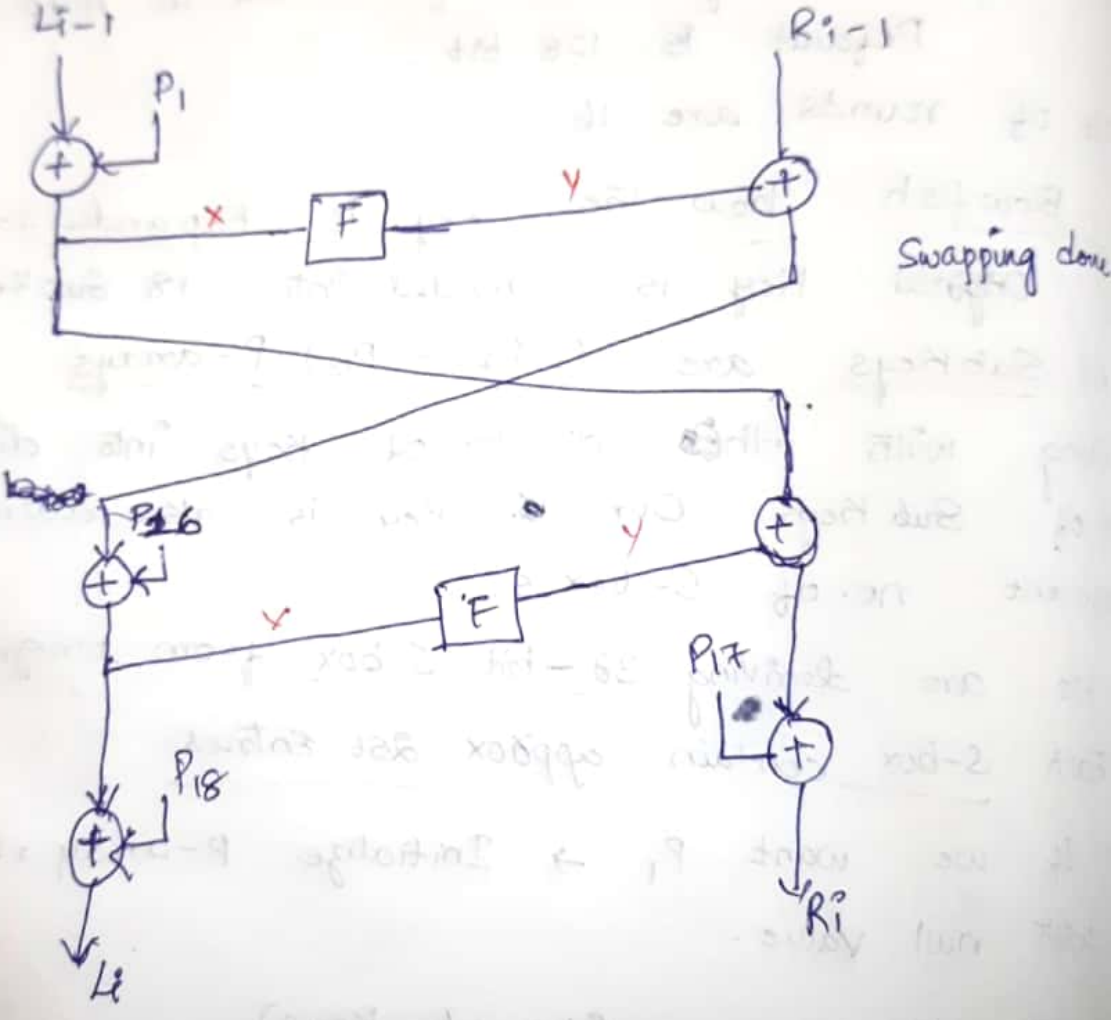
P_{18} XOR K_4

Blowfish:

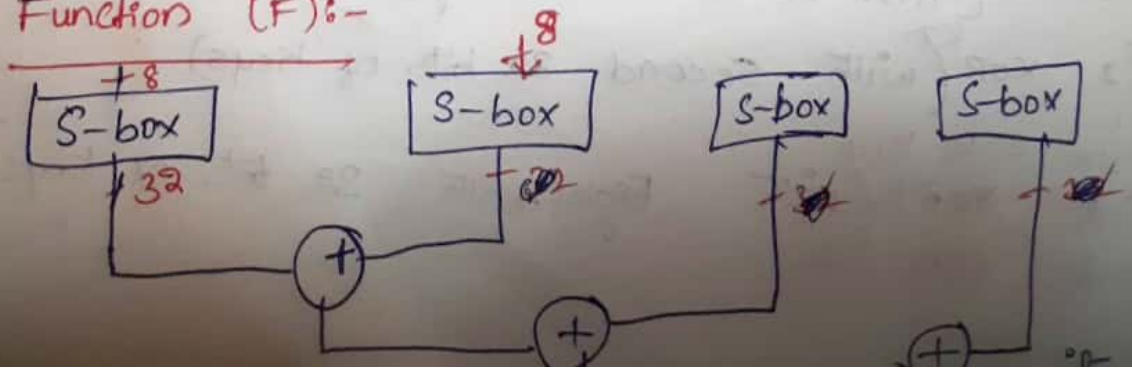
First divide PT into 64 bits into 2 equal halves

- Left 32 bits
- Right 32 bits.

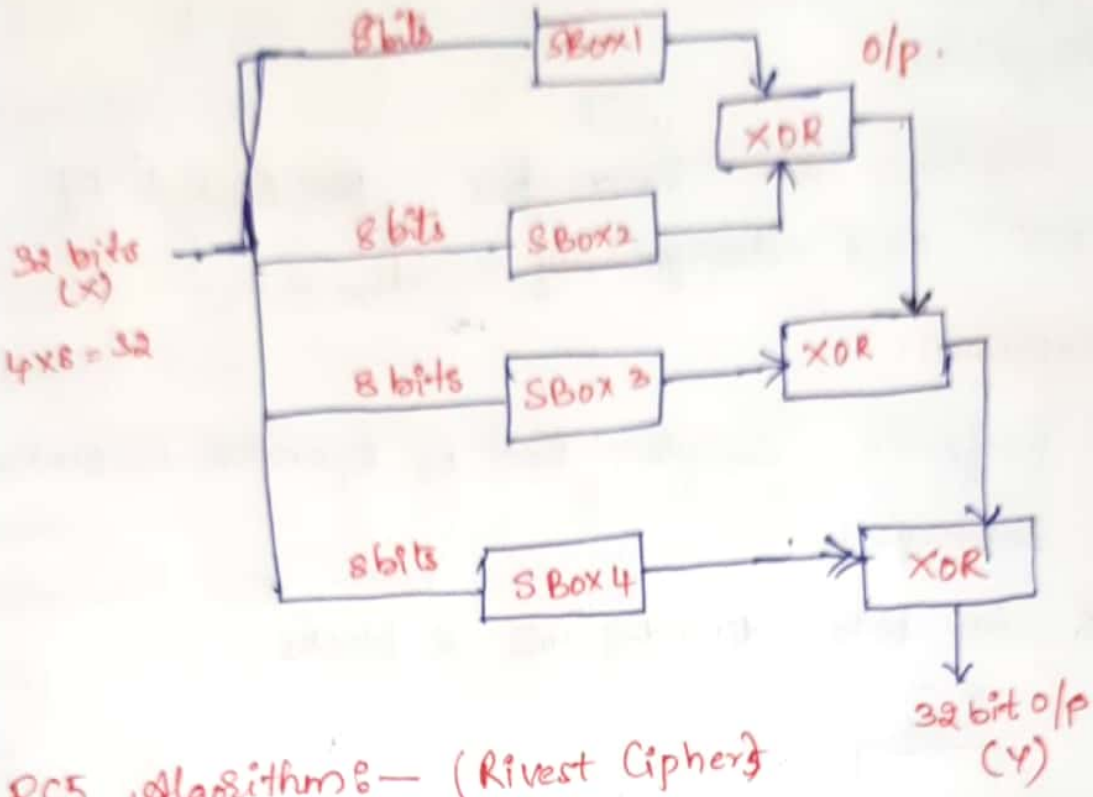
→ After dividing Left & Right we perform XOR operation with P1 Of Left side. After getting result then it is applied to function 'F'. After getting O/p of 'F' it is applied to Right



Function (F):-



→ This algorithm is more secure because the length of size is not known to third party.



RC5 Algorithm:— (Rivest Cipher)

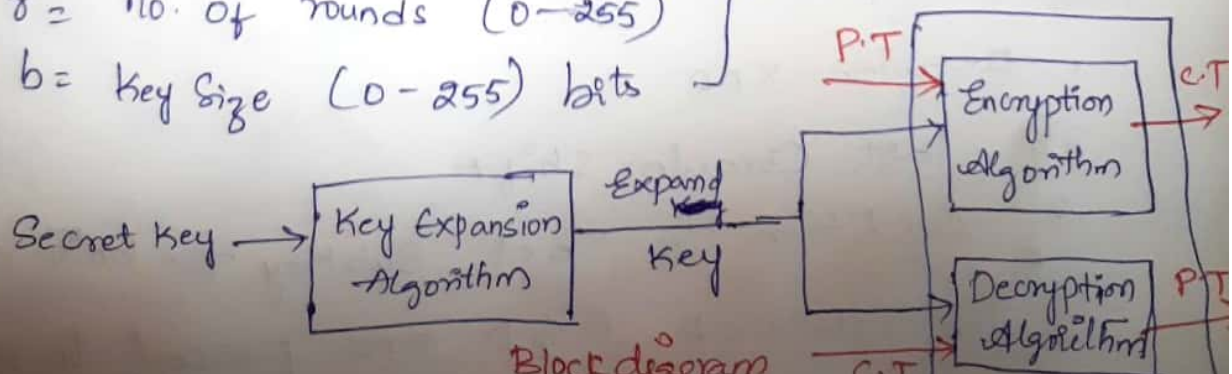
RC5 is a Symmetric Key block encryption algorithm designed by Ron Rivest in 1994.

→ It addresses 2 blocks at a time.

→ PT is divided into 2 blocks.

There are 3 parameters.

w = word size (16, 32, 64 bits) } Based on i/p plain Text.
r = no. of rounds (0–255)
b = key size (0–255) bits



Block diagram

By using Secret Key the Expansion Algorithm Generate Sub Keys.

There are 3 components. They are:

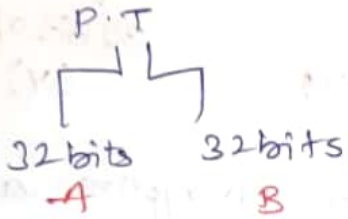
1. Key Expansion
2. Encryption
3. Decryption

These Sub Keys are used for Encrypt^n of the P.T and decrypt^n of the C.T.

1. Key Expansion:

↳ It performs complex set of operation to pr 'B' SubKeys.

↳ PT 64 bits divided into 2 blocks.



Sub Key are generated S[0] and S[1] and these sub keys are added combined to form new blocks. The new blocks are "C & D".

Rounds:

In Each round there are 3 operations to b performed.

1. Bitwise XOR
2. left Circular shift
3. Addition to next SubKey for both C&

The values of C & D are passed to Ne Round. This process will repeat until

we get CT.

Encryption happens in each round:-

We divide the input plain text into two registers A and B each of size w bits. After undergoing the Encryption process the result of A and B together forms the Cipher Text block.

1. One time initialization of PT blocks A & B by adding S[0] & S[1] to A & B respectively. These operations are mod 2^w.
2. XOR A and B . A = A ^ B.
3. Left shift new value of A & B.
4. Add S[2*i] to the O/p of previous step. This is the new value of A.
5. XOR B with new value of A and store in B.
6. Left shift new value of B by A bits.
7. Add S[2*i+1] to the O/p of previous step. This is the new value of B.
8. Repeat entire procedure (except One time Initialization) r times.

```
A = A+S[0]
B = B+S[1]
for i = 1 to r do:
    A = ((A^B) <<< B) + S[2*i]
    B = ((B^A) <<< A) + S[2*i+1]
return A,B
```

RC5, decryption can be defined as:

for i=r down to 1 do:

B ((B-S[2*i+1]) >>> A) ^A

A = ((A-S[2*i]) >>> B) ^ B

B = B - S[1]

A = A - S[0]

return A, B