Tutorial Sheet #03 (Network Security)

T1: Explain the basic structure and detail of one round of DES.

T2: If the plain text 123456ABCD132536 is supplied as input to the DES, what is the initial and inverse initial permutation of this Plain text?

T3: What is IDEA? Explain the sub key generation process for each round of IDEA.

T4: If the plain text in ASCII "COMPUTER" is supplied as input to the DES, what is the initial and inverse initial permutation of this Plain text? (Use the code for ASCII as: C: 01000011, O: 01001111, M: 01001101, P: 01010000, U: 01010101, T: 01010100, E: 01000101, R: 01010010)

T5: For the following hexadecimal data:

AAAA BBBB CCCC DDDD

Find the result after passing it through initial permutation and inverse initial permutation in DES.

T6: What are the four different stages used in the Round 1 of the AES? Explain your answer with example for all the stages.

T7: If the plain text "COMPUTERENGINEER" is supplied as input, then what will be the value of STATE of AES?

T8: Find the value of RCon [11] and RCon[12] constants for the AES-192 and the value of RCon [13] and RCon[14] for AES-256 implementations. Use X^{11-1} mod prime and X^{12-1} mod prime, in which the prime is the irreducible polynomial $(X^8 + X^4 + X^3 + X + 1)$ for AES 192 and use X^{13-1} mod prime and X^{14-1} mod prime, in which the prime is the irreducible polynomial $(X^8 + X^4 + X^3 + X + 1)$ for AES 256

T9: If the value of first, second, third and fourth words in the round of AES is given as W_{00}= 2475A2B3, W_{01}= 34755688, W_{02}= 31E21200 and W_{03}= 13AA5487. Find the value of temporary word (t) used for the round number 1.

T10: What will be values of new states after multiplying the following sates after mix column in AES?

P.T.O

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} * \begin{pmatrix} 87 & F2 & 40 & 97 \\ 6E & 4C & 90 & EC \\ 46 & E7 & 4A & C3 \\ A6 & 8C & D8 & 95 \end{pmatrix}$$

T11: Find the output of Shift rows of the AES after passing the following states as input to the Shift rows:

$$\begin{pmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{pmatrix}$$

T12: Suppose that the round key for round 8 is:

EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F

Then calculate the first 4 bytes (first column) of the round key for round 9.

i (decimal)	temp	After RotWord	After SubWord	Rcon (9)	After XOR with Rcon	w[i- 4]	w[i] = temp XOR w[i- 4]
36						EAD27321	

Soln1:
Explain the basic structure and detail of one round of DES

The Data Encryption Standard (DES) is a symmetric key block cipher that uses a 64-bit key and operates on 64-bit blocks of plaintext. DES works by repeating a series of operations called rounds, with each round involving a combination of substitutions and permutations.

The basic structure of one round of DES is as follows:

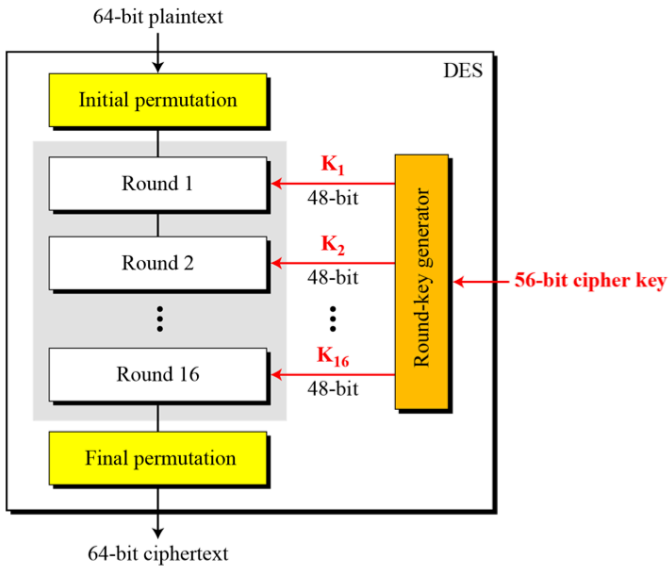
1. Expansion: The 32-bit right half of the previous round's output is expanded to 48 bits using a fixed permutation table called the E-box. The resulting 48-bit block is then XORed with a 48-bit subkey.
2. Substitution: The resulting 48-bit block is divided into eight 6-bit blocks, and each block is substituted using a separate S-box. Each S-box takes a 6-bit input and produces a 4-bit output.
3. Permutation: The 32-bit output of the S-boxes is then subjected to a fixed permutation using a table called the P-box. The P-box shuffles the bits of the output in a non-linear way to produce the final 32-bit output of the round.
4. Key mixing: The output of the P-box is then XORed with the 32-bit left half of the previous round's output to produce the input for the next round.

The subkey used in each round is derived from the 64-bit key using a process called key schedule. The key schedule generates 16 48-bit subkeys, one for each round, by permuting and compressing the original 64-bit key.

The output of the final round is the ciphertext, which is the same length as the plaintext (64 bits). To decrypt the ciphertext, the process is simply reversed by applying the subkeys in reverse order.

Overall, the repeated application of these rounds in DES provides a high level of security, since it involves multiple layers of non-linear transformations that make it difficult to reverse engineer the original key from the ciphertext. However, due to its relatively short key length, DES is no longer considered a secure encryption algorithm and has been superseded by more modern ciphers like AES.

Figure 6.2 *General structure of DES*



DES uses 16 rounds. Each round of DES is a Feistel cipher.

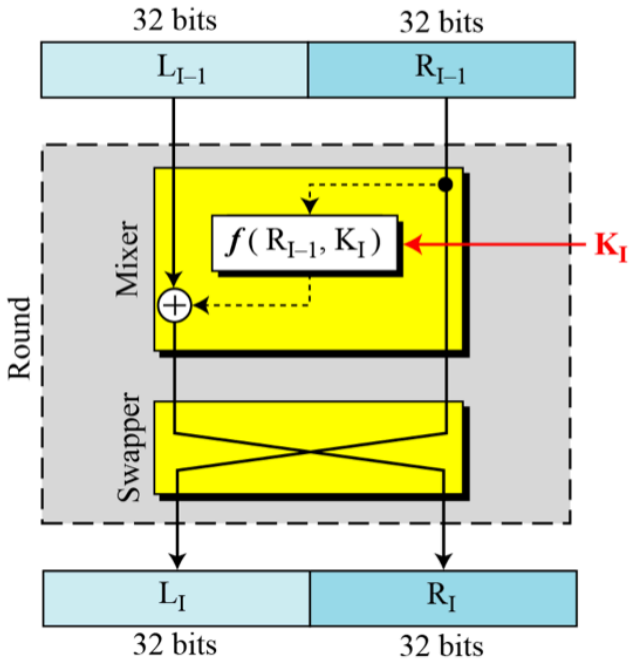


Figure 6.4
A round in DES (encryption site)

The heart of DES is the DES function. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.

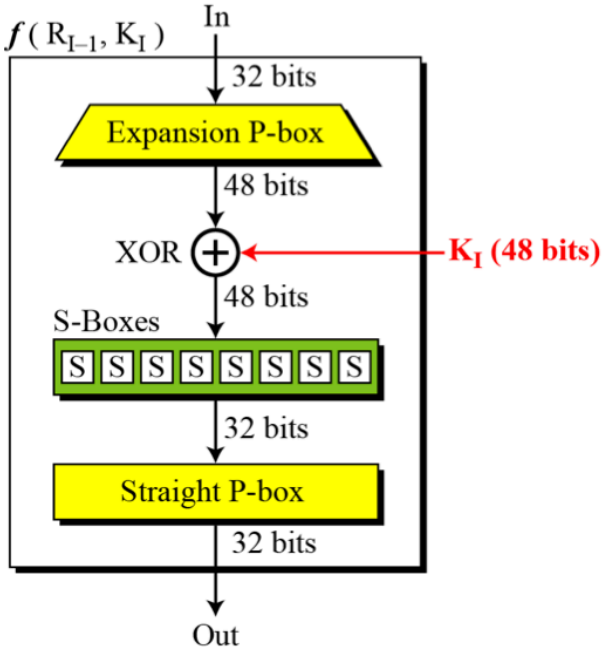


Figure 6.5
DES function

Soln2:

If the plain text 123456ABCD132536 is supplied as input to the DES, what is the initial and inverse initial permutation of this Plain text? **(Final permutation is inverse initial permutation and also the output of initial permutation box)**

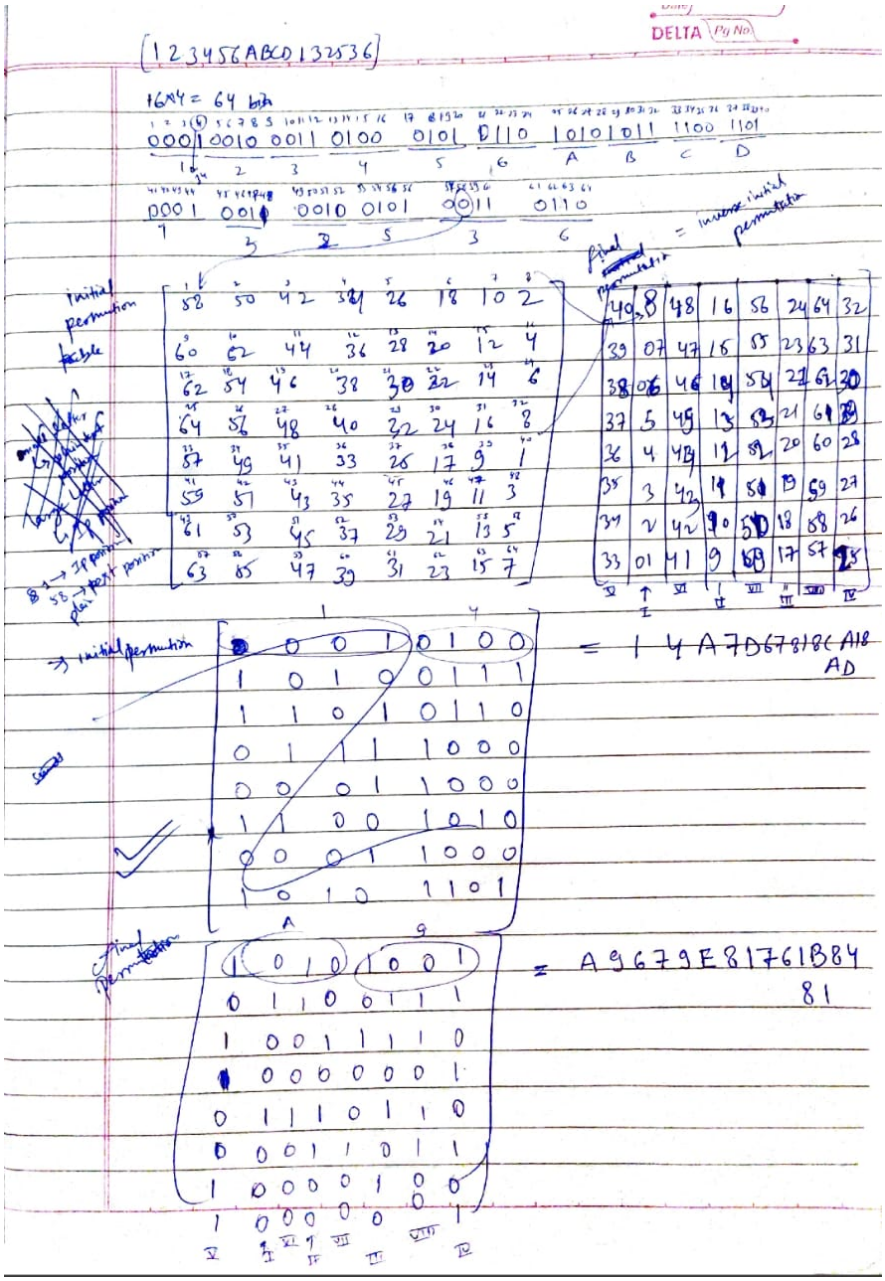
I'm implementing DES, but I cannot understand example.

> **Plaintext**: 123456ABCD132536
> **After initial permutation**: 14A7D67818CA18AD

Plaintext is 64-bits hexadecimal, so each character is 4-bits. The first entry of IP(Initial Permutation) table is 58 which means 58-th binary number in the plaintext.

The DES specification numbers bits form 1 to 64, in reading order of big-endian data.

Bit 58 belongs to the second rightmost hex digit of the plaintext, which contains bits 57 to 60. This hex digit is `3`, that is `0011` in binary, and bit 58 is the second `0` in that.



Soln3:

What is IDEA? Explain the sub key generation process for each round of IDEA.

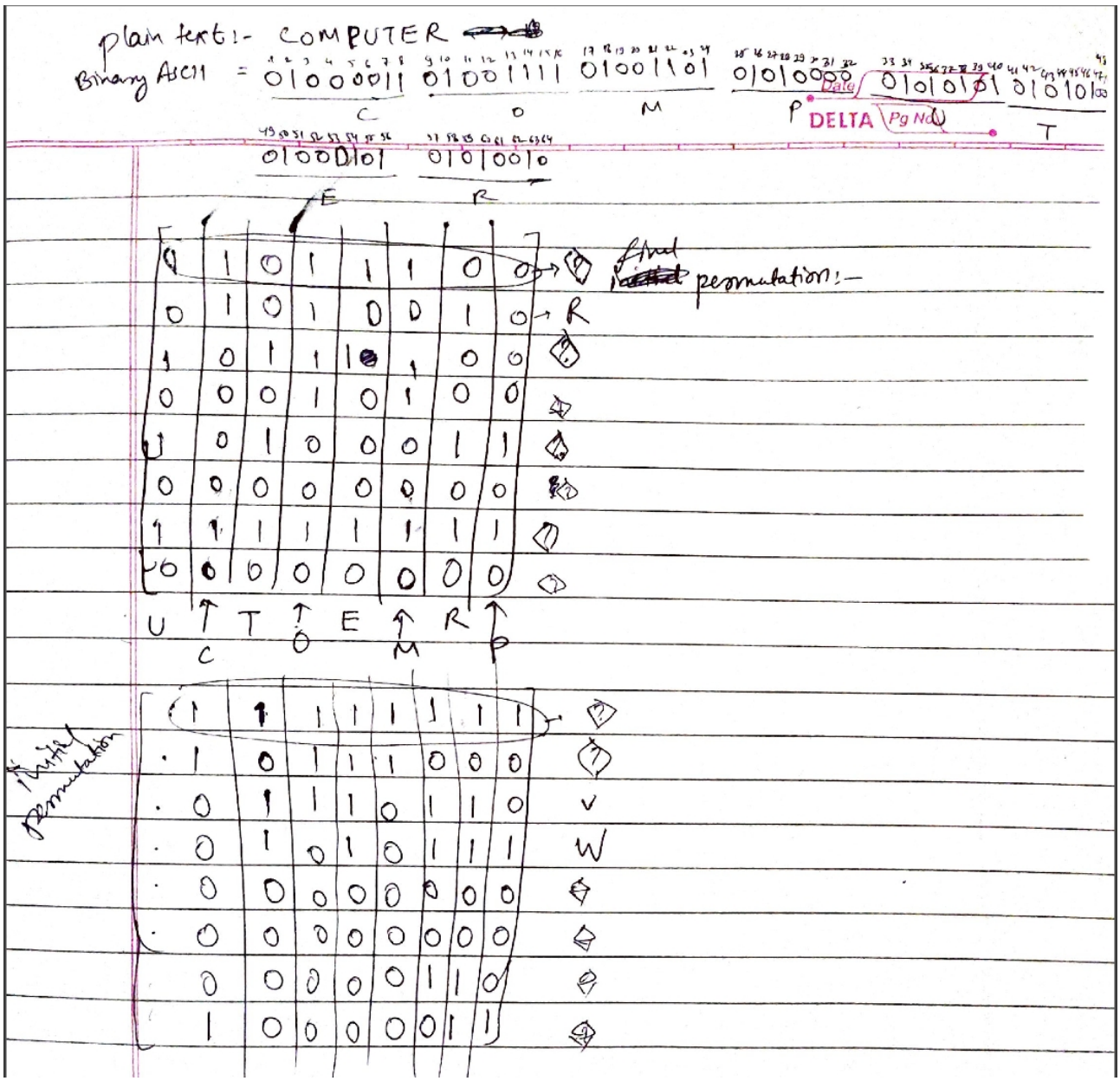
IDEA (International Data Encryption Algorithm) is a symmetric key block cipher that was developed in 1991 as a replacement for the aging DES algorithm. IDEA uses a 128-bit key and operates on 64-bit blocks of data.

The sub key generation process for each round of IDEA is as follows:

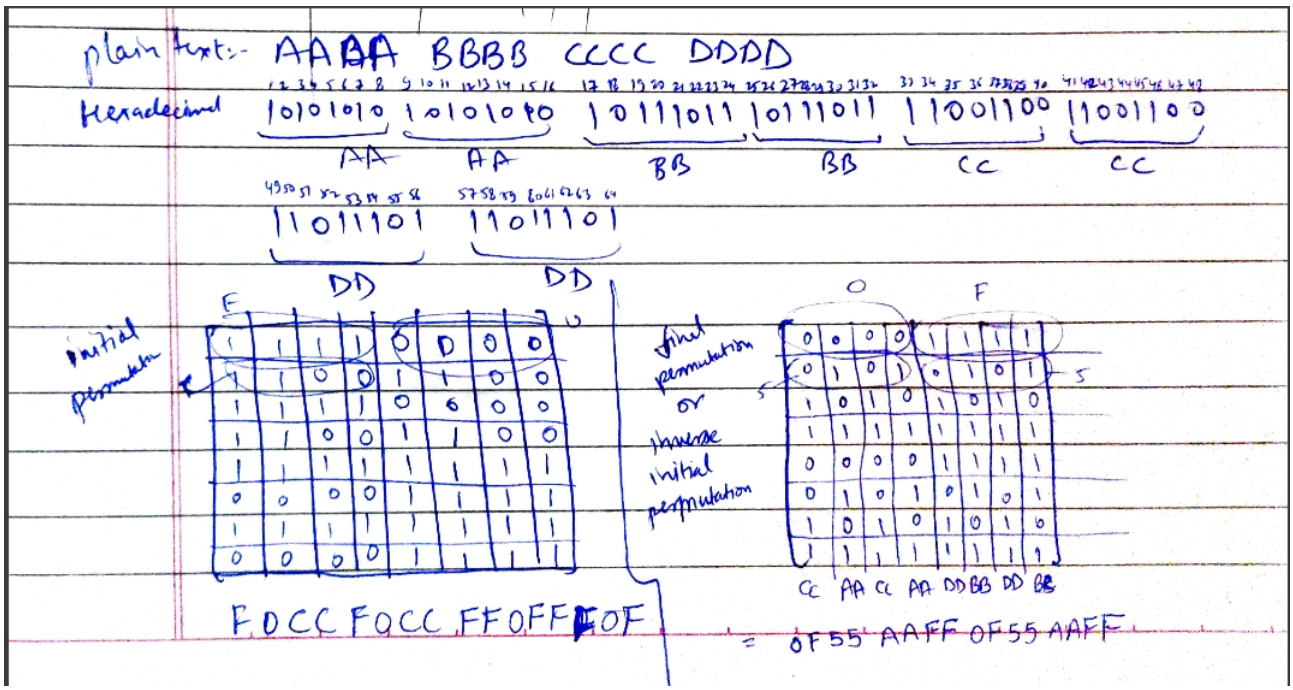
1. Key Expansion:
2. The 128-bit IDEA key is split into eight 16-bit subkeys. These subkeys are then used to generate 52 additional 16-bit subkeys that will be used in the encryption and decryption rounds.
3. Encryption/Decryption Rounds:
4. IDEA uses 8 rounds of encryption/decryption on the input block. In each round, the input block is divided into two 16-bit halves: the left half (L) and the right half (R). The key material used in each round is derived from the 56 subkeys generated during the Key Expansion phase.
   The following steps are performed in each round:
5. a. Multiply L by a 16-bit subkey (K1-K6).
6. b. Add R to the result of step (a).
7. c. Modulo 2^16 (keep the lower 16 bits).
8. d. Multiply the result of step (c) by a 16-bit subkey (K7-K12).
9. e. Add the result of step (d) to L.
10. f. Modulo 2^16 (keep the lower 16 bits).
11. g. Exchange L and R.
    After 8 rounds, the L and R halves are concatenated and passed through the final permutation to obtain the ciphertext.

The sub key generation process for IDEA is a critical part of the cipher, as it ensures that each round has a unique key material. This helps to ensure the security of the algorithm by preventing attacks that exploit the reuse of key material. The use of multiple rounds and a complex key schedule also contribute to the strength of IDEA as a block cipher.

Soln4:
If the plain text in ASCII "COMPUTER" is supplied as input to the DES, what is initial and inverse initial permutation of this Plain text?



Soln5:
If the plain text in ASCII "COMPUTER" is supplied as input to the DES, what is initial and inverse initial permutation of this Plain text?



Soln6:
What are the four different stages used in the Round 1 of the AES? Explain your answer with example for all the stages.

The Advanced Encryption Standard (AES) is a symmetric key block cipher that operates on 128-bit blocks of data. AES consists of multiple rounds of encryption, with each round consisting of four different stages: SubBytes, ShiftRows, MixColumns, and AddRoundKey.

The stages of Round 1 of AES are as follows:

1. SubBytes: In this stage, each byte of the input block is substituted with a corresponding byte from the S-box. The S-box is a fixed 256-byte table that is generated during the key expansion phase of the cipher. The substitution is performed based on the values of each byte, with the same input byte always being replaced by the same output byte.
   Example:

Suppose the input block is represented as a 4x4 matrix of bytes, where each byte is represented in hexadecimal notation. The following is an example input block:

19 3D E3 BE
A0 F4 E2 2B
9A C6 8D 2A
E9 F8 48 08

In the SubBytes stage, each byte is replaced with a corresponding byte from the S-box. The following is an example output block:

D4 27 11 AE
E0 BF 98 F1
B8 B4 5D E5
1E 41 52 30

2. ShiftRows: In this stage, the rows of the input block are shifted cyclically by a certain number of bytes. The number of shifts for each row is determined by the row number. The first row is not

shifted, the second row is shifted by one byte, the third row is shifted by two bytes, and the fourth row is shifted by three bytes.

Example:

Using the output block from the SubBytes stage, the ShiftRows stage performs the following cyclic shifts on each row:

D4 27 11 AE D4 27 11 AE
E0 BF 98 F1 BF 98 F1 E0
B8 B4 5D E5 5D E5 B8 B4
1E 41 52 30 30 1E 41 52

3. MixColumns: In this stage, each column of the input block is transformed using a fixed matrix multiplication. This transformation provides diffusion by mixing the input bits across different bytes.

   Example:

Using the output block from the ShiftRows stage, the MixColumns stage performs a fixed matrix multiplication on each column. The following is the resulting output block:

04 A0 8E D5
88 52 2B 2F
D2 7F 8D D6
BC 9E 17 B8

4. AddRoundKey: In this stage, the input block is combined with a round key. The round key is generated from the cipher key using a key schedule algorithm. The round key is added to the input block using bitwise XOR.

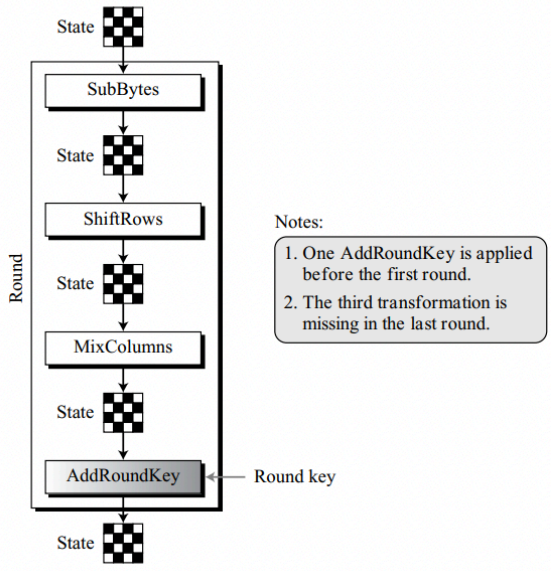
   Example:

Using the output block from the MixColumns stage, the AddRoundKey stage combines the input block with a round key. The round key is generated from the cipher key and XORed with the input block. The following is the resulting output block:

04 2A 6C 78
46 F2 F9 9B
DA F4 C7 C2
5D 50 13 C8

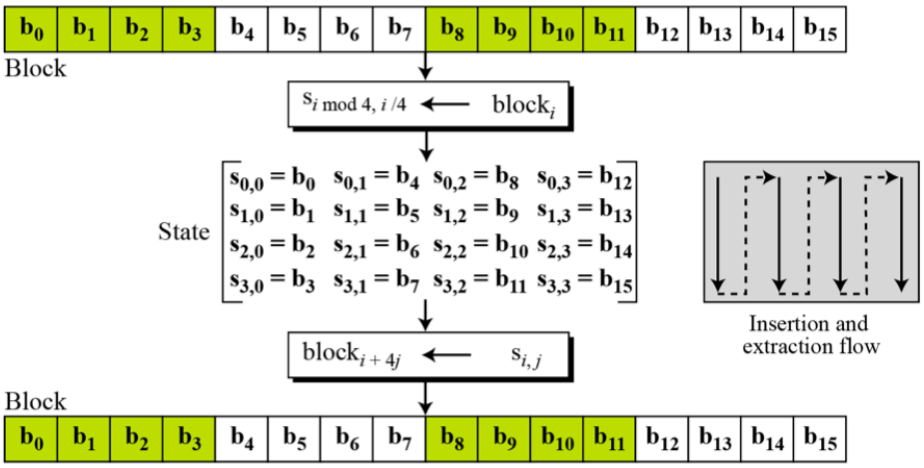
Overall, the four stages of Round 1 of AES work together to provide strong security for the cipher. The SubBytes and ShiftRows stages provide confusion by substituting and rearranging.

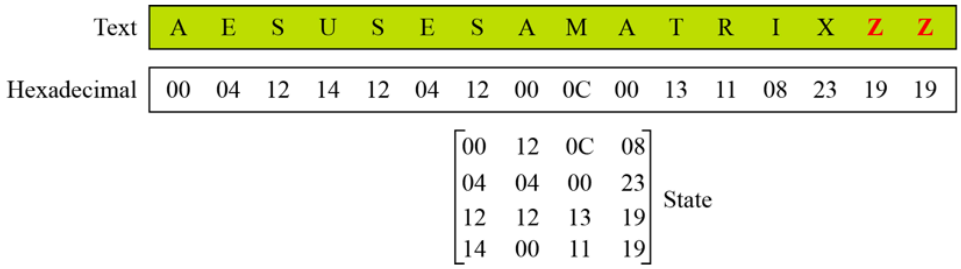
Figure 7.5 *Structure of each round at the encryption site*



Soln7:
If the plain text "COMPUTERENGINEER" is supplied as input, then what will be the value of STATE of AES.

Figure 7.3 *Block-to-state and state-to-block transformation*





Plain text:
Hexadecimal:

C	O	M	P	U	T	E	R	E	N	G	I	N	E	E	R
2	E	C	F	14	13	4	11	4	D	6	8	D	4	4	11

STATE:

2	14	4	D
E	13	D	4
C	4	6	4
F	11	8	11

Soln8:
Find the value of RCon[11] and RCon[12] constants for AES-192 and the value of RCon[13] and RCon[14] for AES-256 implementations. Use x^{11-1} mod prime and x^{12-1} mod prime, in which prime is the irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$) for AES 192 and use x^{13-1} mod prime and x^{14-1} mod prime, in which prime is the irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$) for AES 256.

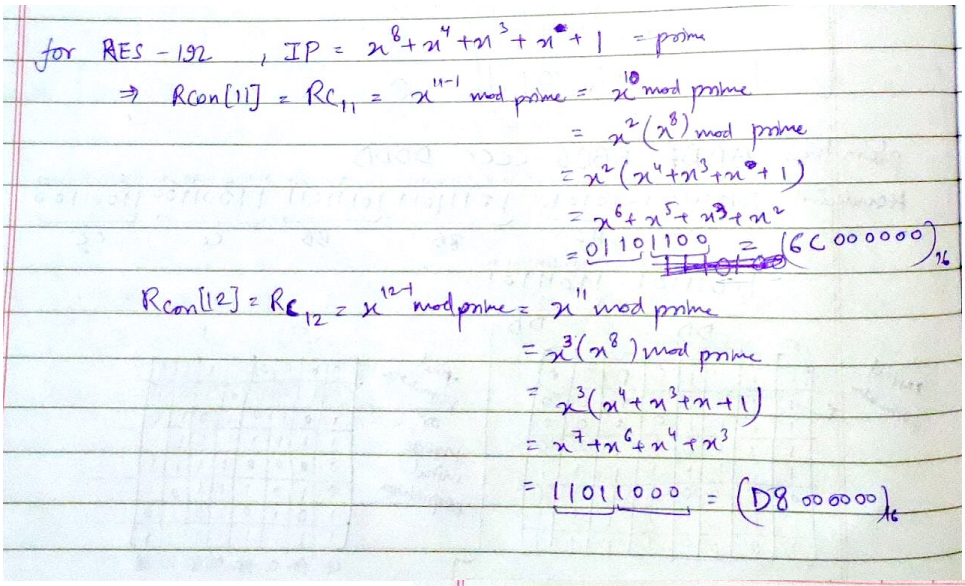
21.

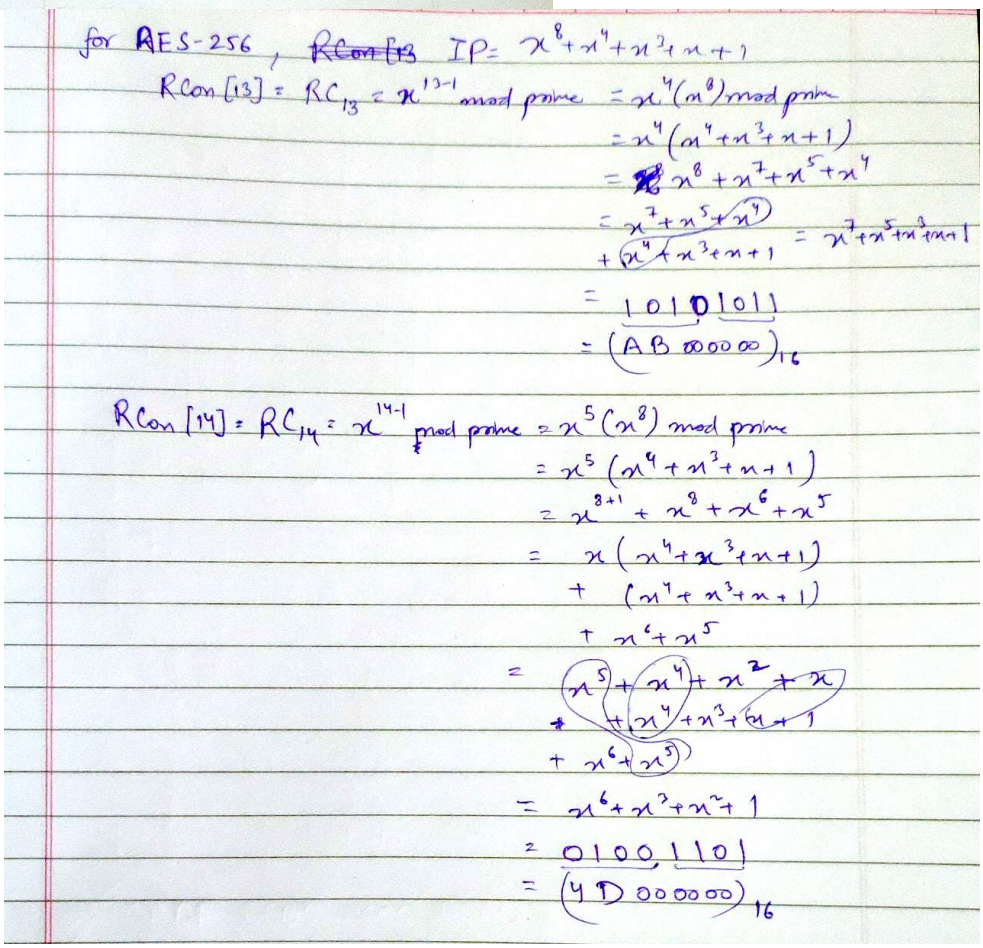
a. We can use (x^{11-1} mod prime) and (x^{12-1} mod prime), in which the prime is the the irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$), to find the first terms of RCon[11] and RCon[12]. The following shows the constants for AES-192.

Round	*(RCon)*	*Round*	*(RCon)*	*Round*	*(RCon)*
1	(**01** 00 00 00)$_{16}$	5	(**10** 00 00 00)$_{16}$	9	(**1B** 00 00 00)$_{16}$
2	(**02** 00 00 00)$_{16}$	6	(**20** 00 00 00)$_{16}$	10	(**36** 00 00 00)$_{16}$
3	(**04** 00 00 00)$_{16}$	7	(**40** 00 00 00)$_{16}$	11	(**6C** 00 00 00)$_{16}$
4	(**08** 00 00 00)$_{16}$	8	(**80** 00 00 00)$_{16}$	12	(**D8** 00 00 00)$_{16}$

b. We can use (x^{13-1} mod prime) and (x^{14-1} mod prime), in which the prime is the the irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$), to find the first terms of RCon[13] and RCon[14]. The following shows the constants for AES-256.

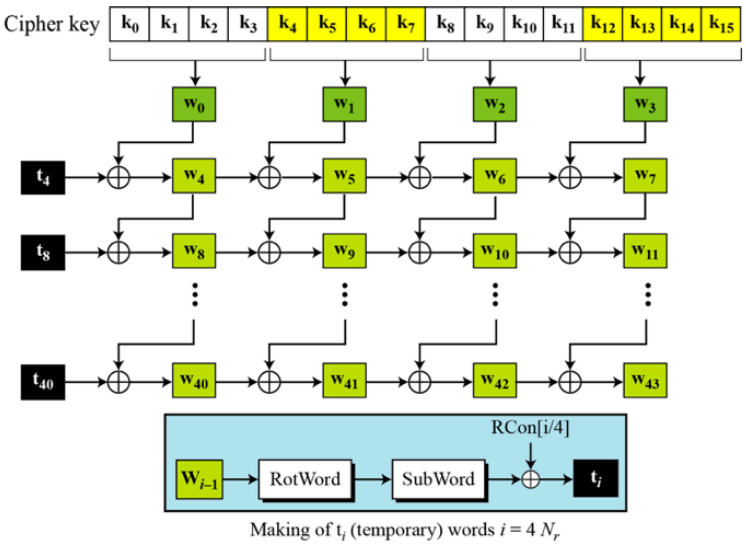
Round	*(RCon)*	*Round*	*(RCon)*	*Round*	*(RCon)*
1	(**01** 00 00 00)$_{16}$	6	(**20** 00 00 00)$_{16}$	11	(**6C** 00 00 00)$_{16}$
2	(**02** 00 00 00)$_{16}$	7	(**40** 00 00 00)$_{16}$	12	(**D8** 00 00 00)$_{16}$
3	(**04** 00 00 00)$_{16}$	8	(**80** 00 00 00)$_{16}$	13	(**AB** 00 00 00)$_{16}$
4	(**08** 00 00 00)$_{16}$	9	(**1B** 00 00 00)$_{16}$	14	(**4D** 00 00 00)$_{16}$
5	(**10** 00 00 00)$_{16}$	10	(**36** 00 00 00)$_{16}$		





Soln9:
if the value of first , second, third and fourth words in the round of AES is given as W_{00}=2475A2B3 W_{01}=34755688, W_{02}=31E21200 and W_{03}=13AA5487. Find the value of temporary word(t) used for the first round number 1.

Figure 7.16 *Key expansion in AES*



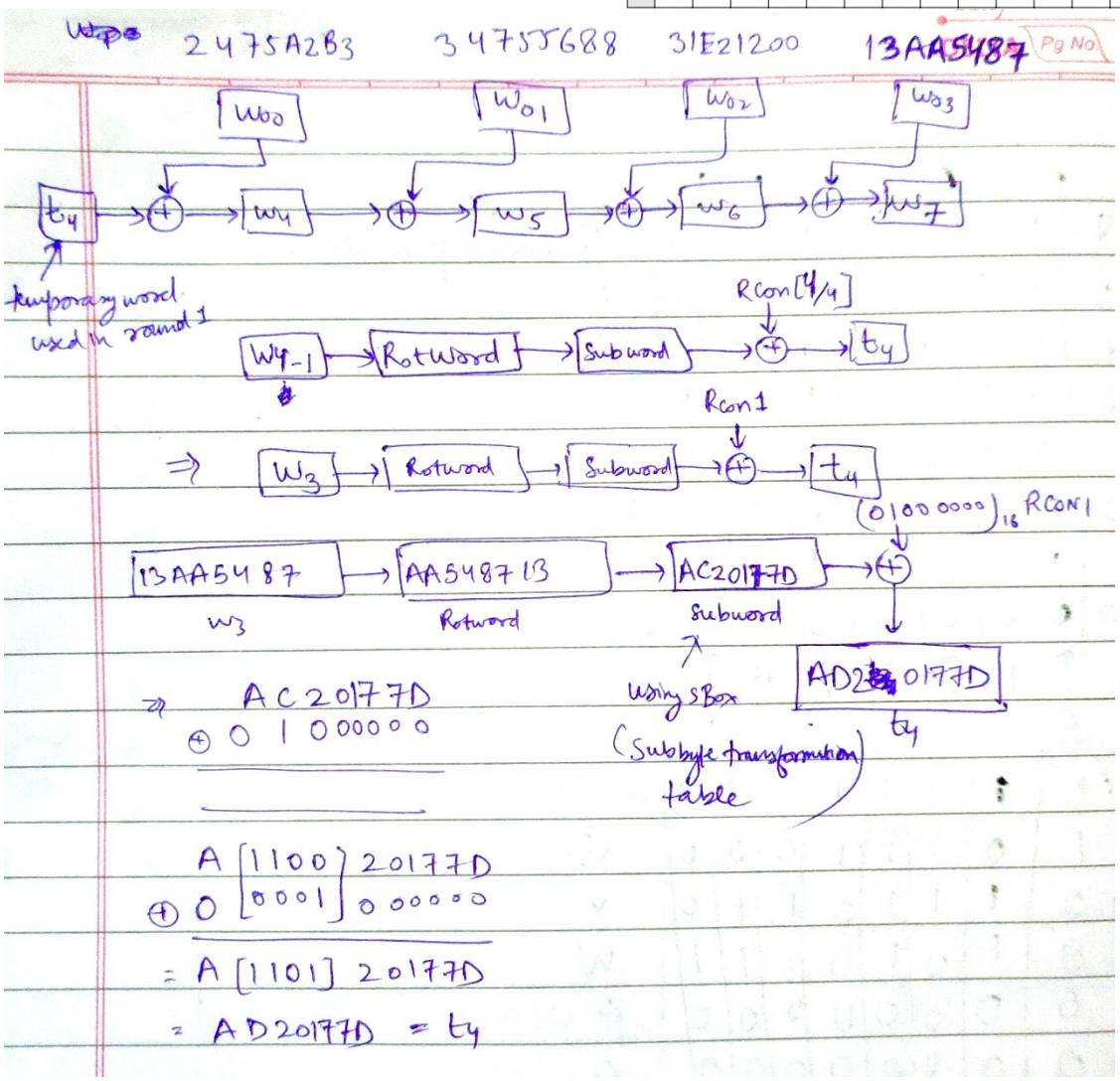
Making of t_i (temporary) words $i = 4\ N_r$

Table 7.1 *SubBytes transformation table*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8

Table 7.1 *SubBytes transformation table (continued)*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	CB	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16



Soln10:
What will be the values of new states after multiplying the following states after mix column in AES?

MixColumns Transformation

Forward and Inverse Transformations The **forward mix column transformation**, called MixColumns, operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column. The transformation can be defined by the following matrix multiplication on **State** (Figure 5.5b):

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \quad (5.3)$$

Each element in the product matrix is the sum of products of elements of one row and one column. In this case, the individual additions and multiplications[6] are performed in GF(2^8). The MixColumns transformation on a single column $j (0 \le j \le 3)$ of **State** can be expressed as

$$\begin{aligned} s'_{0,j} &= (2 \cdot s_{0,j}) \oplus (3 \cdot s_{1,j}) \oplus s_{2,j} \oplus s_{3,j} \\ s'_{1,j} &= s_{0,j} \oplus (2 \cdot s_{1,j}) \oplus (3 \cdot s_{2,j}) \oplus s_{3,j} \\ s'_{2,j} &= s_{0,j} \oplus s_{1,j} \oplus (2 \cdot s_{2,j}) \oplus (3 \cdot s_{3,j}) \\ s'_{3,j} &= (3 \cdot s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \cdot s_{3,j}) \end{aligned} \quad (5.4)$$

The following is an example of MixColumns:

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

Let us verify the first column of this example. Recall from Section 4.6 that, in GF(2^8), addition is the bitwise XOR operation and that multiplication can be performed according to the rule established in Equation (4.10). In particular, multiplication of a value by x (i.e., by {02}) can be implemented as a 1-bit left shift followed by a conditional bitwise XOR with (0001 1011) if the leftmost bit of the original value (prior to the shift) is 1. Thus, to verify the MixColumns transformation on the first column, we need to show that

$$\begin{aligned} (\{02\} \cdot \{87\}) \oplus (\{03\} \cdot \{6E\}) \oplus \{46\} \oplus \{A6\} &= \{47\} \\ \{87\} \oplus (\{02\} \cdot \{6E\}) \oplus (\{03\} \cdot \{46\}) \oplus \{A6\} &= \{37\} \\ \{87\} \oplus \{6E\} \oplus (\{02\} \cdot \{46\}) \oplus (\{03\} \cdot \{A6\}) &= \{94\} \\ (\{03\} \cdot \{87\}) \oplus \{6E\} \oplus \{46\} \oplus (\{02\} \cdot \{A6\}) &= \{ED\} \end{aligned}$$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & & & \\ S_{2,0} & & & \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

Predefine matrix — State array

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} 87 & F2 & 40 & 97 \\ 6E & 4C & 90 & EC \\ 46 & E7 & 4A & 03 \\ A6 & 8C & D8 & 95 \end{bmatrix}$$

$$= \begin{bmatrix} 47 & 40 & A3 & 4C \\ 37 & D4 & 70 & 9F \\ 94 & E4 & 3A & 42 \\ ED & A5 & A6 & BC \end{bmatrix}$$

$(02 \times 87) \oplus (03 \times 6E) \oplus (01 \times 46) \oplus (01 \times A6) = 47$

$02 = 00000010 = x$

$87 = 10000111 = x^7 + x^2 + x + 1$ $\quad x^8 = x^4 + x^3 + x + 1$

$x(x^7 + x^2 + x + 1) = x^8 + x^3 + x^2 + x$

$= x^4 + x^3 + x + 1$
$+ x^3 + x^2 + x$

$= x^4 + x^2 + 1 = 00010101 = 15$

$03 = 00000011 = x + 1$

$6E = 01101110 = x^6 + x^5 + x^3 + x^2 + x$

$03 \times 6E = (x+1)(x^6 + x^5 + x^3 + x^2 + x)$

$= x^7 + x^6 + x^4 + x^3 + x^2$
$x^6 + x^5 + x^3 + x^2 + x$

$= x^7 + x^5 + x^4 + x$

$= 10110010 = B2$

$01 \times 46 = 46 = 01000110$

$01 \times A6 = A6 = 10100110$

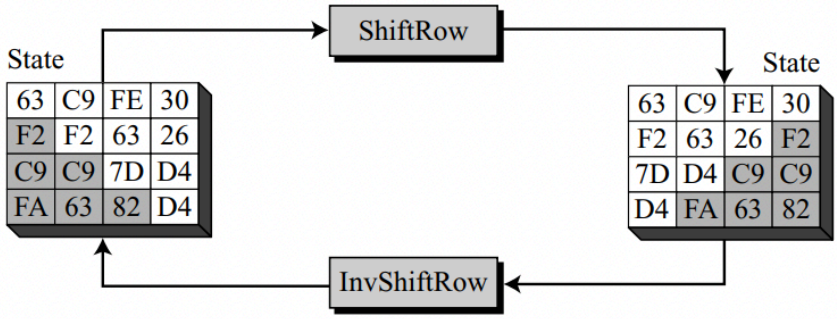
B2 — 10110010
46 — 01000110
A6 — 10100110
47 — 01000111

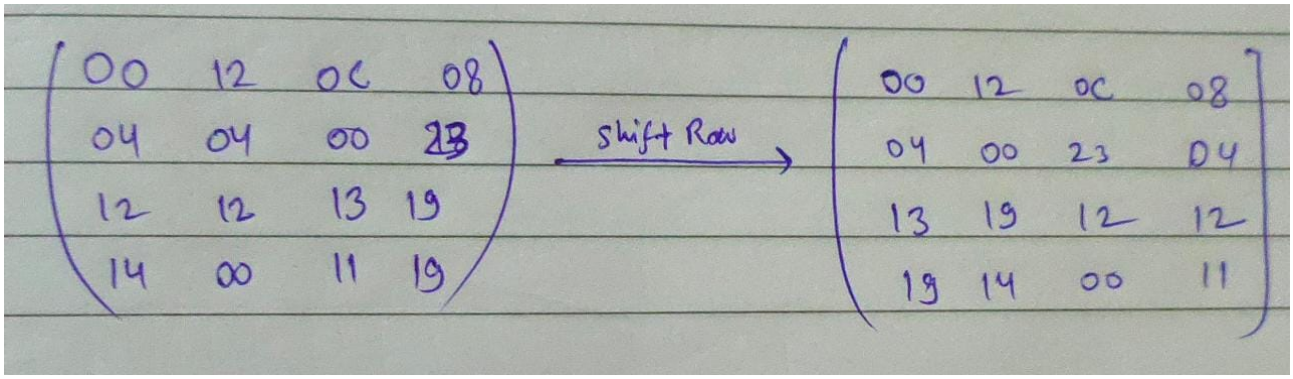
Soln 11:
Find the output of the Shift rows of the aes after passing the following state as input to the Shift rows.

Algorithm 7.2 *Pseudocode for ShiftRows transformation*

```
ShiftRows (S)
{
    for (r = 1 to 3)
        shiftrow (s_r, r)                    // s_r is the rth row
}
shiftrow (row, n)                            // n is the number of bytes to be shifted
{
   CopyRow (row, t)                          // t is a temporary row
   for (c = 0 to 3)
        row_((c – n) mod 4) ← t_c
}
```

Figure 7.10 *ShiftRows transformation in Example 7.4*



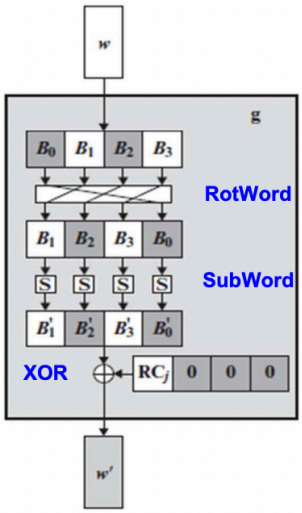


Soln12:

AES Key Expansion

The round constant is different for each round and is defined as Rcon[j] = (RC[j], 0, 0, 0), with RC[1] = 1, RC[j] = 2 · RC[j - 1].
The values of RC[j] in hexadecimal are:

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36



For example: Suppose that the round key for round 8 is
EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F
Then the first 4 bytes (first column) of the round key for round 9 are calculated as follows:

i (decimal)	temp	After RotWord	After SubWord	Rcon (9)	After XOR with Rcon	w[i−4]	w[i] = temp ⊕ w[i−4]
36	7F8D292F	8D292F7F	5DA515D2	1B000000	46A515D2	EAD27321	AC7766F3

Operations on g function
(1) RotWord performs a one-byte circular left shift on a word
(2) SubWord performs a byte substitution on each byte of its input word using the S-box
(3) XOR W_i' xor $[RC_j, 0, 0, 0]$