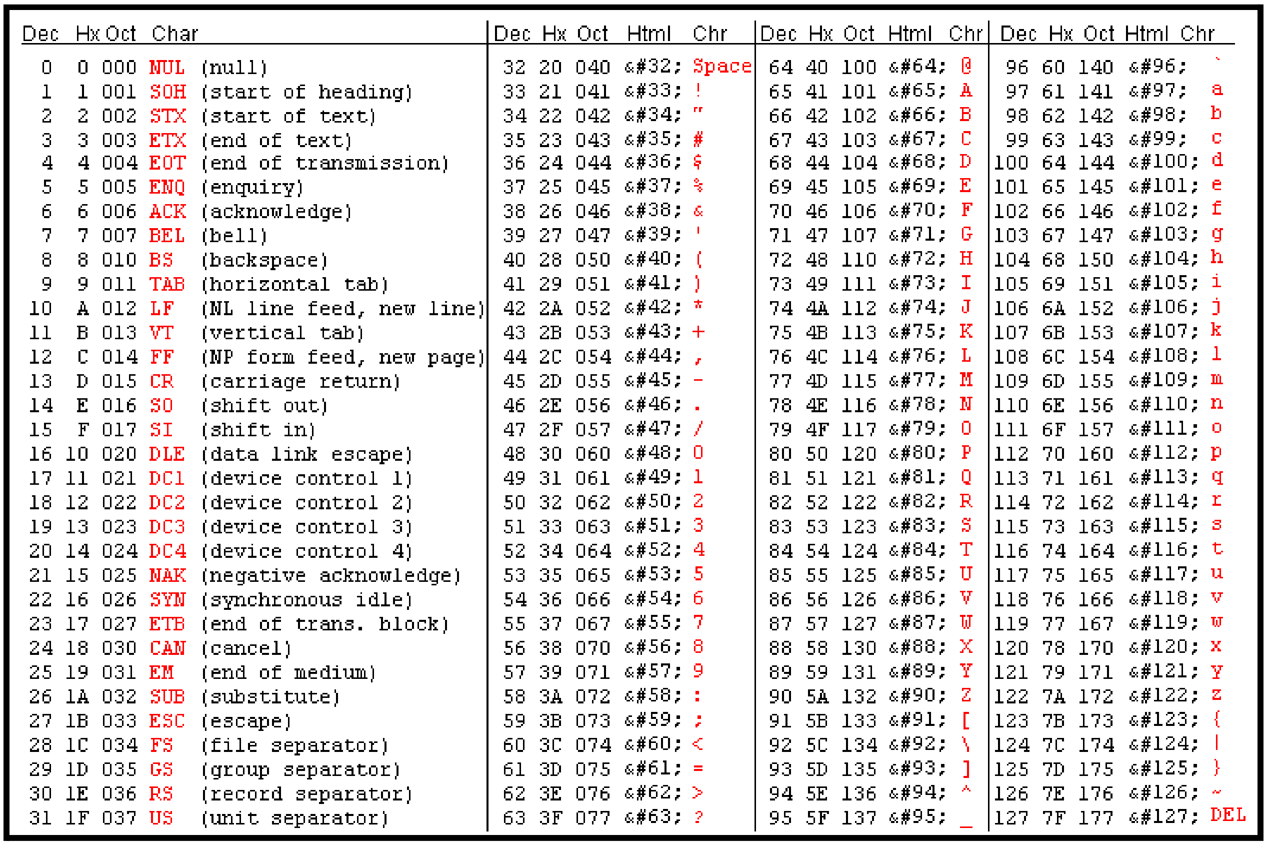
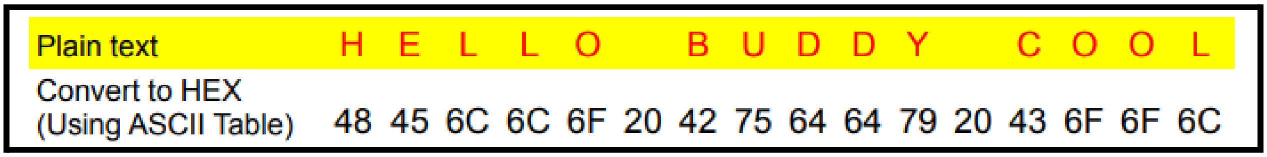
Convert plain text into HEX using ASCII Value



Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH	(start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX	(start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX	(end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO	(shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI	(shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE	(data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1	(device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2	(device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3	(device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4	(device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN	(cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM	(end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB	(substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC	(escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS	(file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS	(group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS	(record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US	(unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

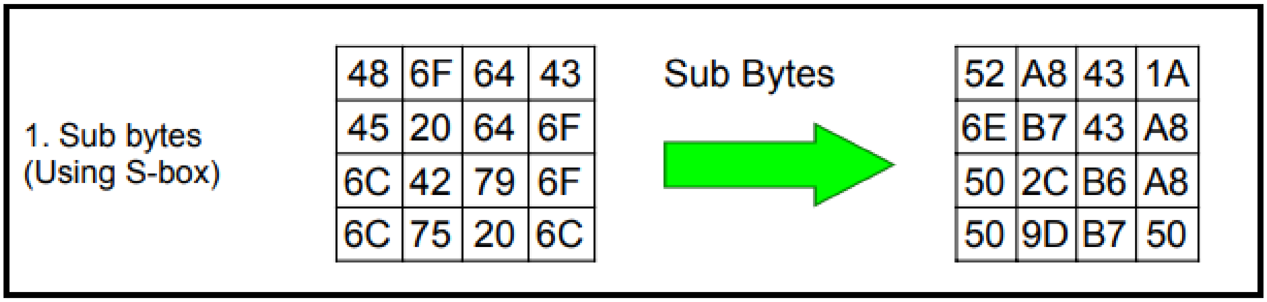


Plain text	H	E	L	L	O		B	U	D	D	Y		C	O	O	L
Convert to HEX (Using ASCII Table)	48	45	6C	6C	6F	20	42	75	64	64	79	20	43	6F	6F	6C

Step 1 : Sub bytes

S-box :

	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
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	C8	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



1. Sub bytes
(Using S-box)

48	6F	64	43
45	20	64	6F
6C	42	79	6F
6C	75	20	6C

Sub Bytes

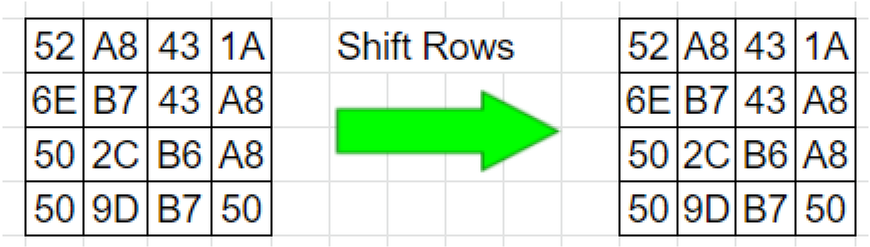
52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

E.g 48

1. 4 for x & 8 for y
   = 52

Step 2: Shift Rows (Left rotate each row by 0,1,2,3 bytes respectively)

1st row : No need to shift

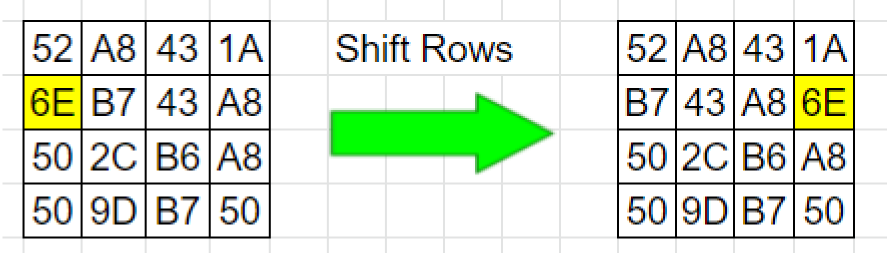


52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

Shift Rows

52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

2nd row : Shift 1 byte

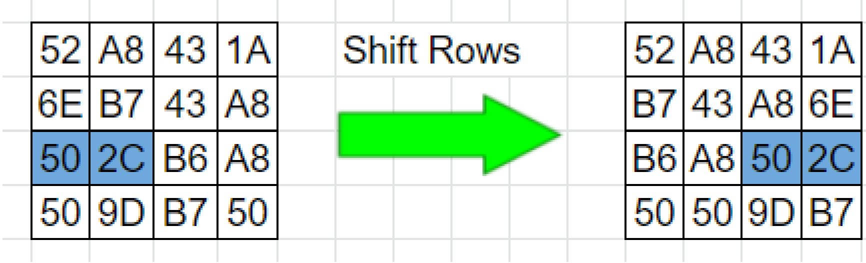


52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

Shift Rows

52	A8	43	1A
B7	43	A8	6E
50	2C	B6	A8
50	9D	B7	50

3rd row : Shift 2 byte

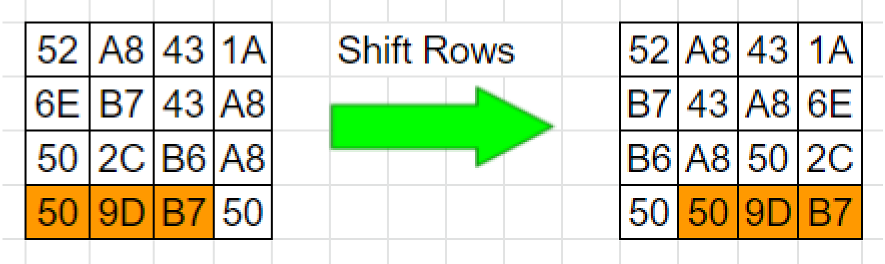


52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

Shift Rows

52	A8	43	1A
B7	43	A8	6E
B6	A8	50	2C
50	50	9D	B7

4th row : Shift 3 byte



52	A8	43	1A
6E	B7	43	A8
50	2C	B6	A8
50	9D	B7	50

Shift Rows

52	A8	43	1A
B7	43	A8	6E
B6	A8	50	2C
50	50	9D	B7

Step 3 : Matrix Multiplication : row * column

Notes for mix column

1. The multiplication of a value by 02 can be obtained :
    a. Convert it into binary
    b. 1-bit left shift followed by a conditional bitwise xor with (00011011) if the leftmost bit of the original value (before the shift) is 1
    c. Pad the binary value with 0's if not enough 8 bits

2. Split 03 up in its binary form as:
   = {03}
   = {10 XOR 01}

Useful Website for calculating the mix column

1. Shift bit
   Bit Shift Calculator

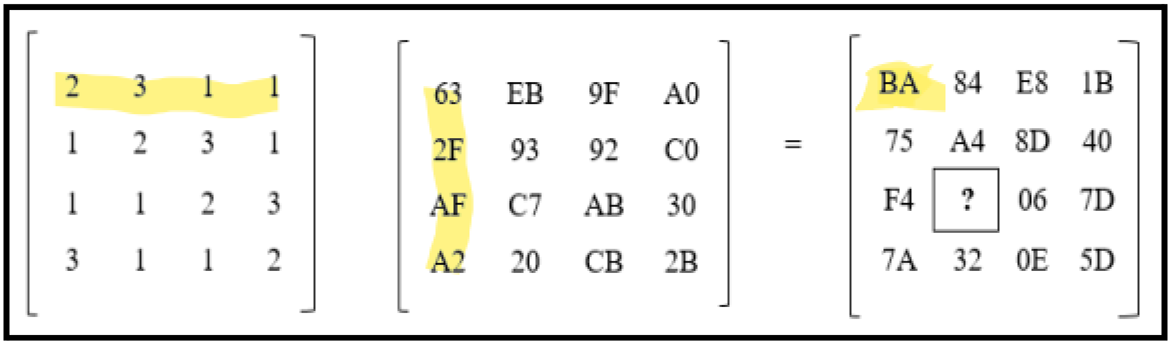
2. Hex to binary
   Hexadecimal to Decimal Converter - Conversion

3. Binary calculator (+, -, *, /)
   https://www.calculator.net/binary-calculator.htm

4. XOR calculator
   XOR Calculator

5. Hex calculator
   Hex Calculator

Example 1 : Calculation for 1st row 1st column :



$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 63 & EB & 9F & A0 \\ 2F & 93 & 92 & C0 \\ AF & C7 & AB & 30 \\ A2 & 20 & CB & 2B \end{bmatrix} = \begin{bmatrix} BA & 84 & E8 & 1B \\ 75 & A4 & 8D & 40 \\ F4 & \boxed{?} & 06 & 7D \\ 7A & 32 & 0E & 5D \end{bmatrix}$$

1. **{02.63}**

   63 = 01100011

   {63} . {02}

   = 01100011 << 1

   = 11000110

2. **{03.2f}**

   2F = 00101111

   {03} . {2F}

   = {10 XOR 01} . {00101111}

   = {00101111 . 10} XOR {00101111 . 01}

   = {00101111 . 10} XOR {00101111}

   = 01011110 XOR 00101111

   = 01110001

3. **{01.AF}**

   AF = 10101111

4. **{01.A2}**

   A2 = 10100010

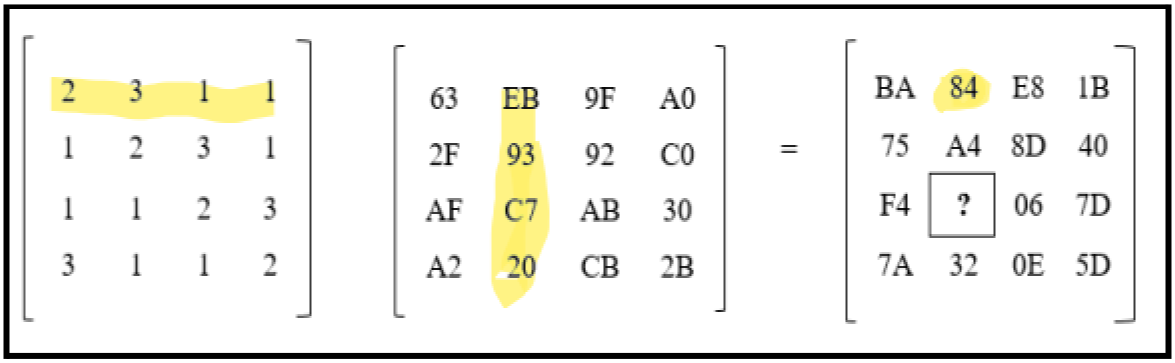
Answer:

= {02.63} + {03.2f} + {01.AF} + {01.A2}

= 11000110 XOR 01110001 XOR 10101111 XOR 10100010

= 10111010 (BA in hex)

Example 2 : Calculation for 1st row 2nd column



$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 63 & EB & 9F & A0 \\ 2F & 93 & 92 & C0 \\ AF & C7 & AB & 30 \\ A2 & 20 & CB & 2B \end{bmatrix} = \begin{bmatrix} BA & 84 & E8 & 1B \\ 75 & A4 & 8D & 40 \\ F4 & \boxed{?} & 06 & 7D \\ 7A & 32 & 0E & 5D \end{bmatrix}$$

1. **{02 . EB}**
   EB = 11101011
   = {EB}. {02}
   = 11101011 << 1
   = 11010110
   = 11010110 XOR 00011011
   = 11001101

2. **{03 . 93}**
   93 = 10010011
   = {03} . {93}
   = {10 XOR 01} . {10010011}
   = {10010011 . 10} XOR {10010011 . 01}
   = {10010011 . 10} XOR {10010011}
   = 00100110 XOR 00011011 XOR 10010011
   = 10101110

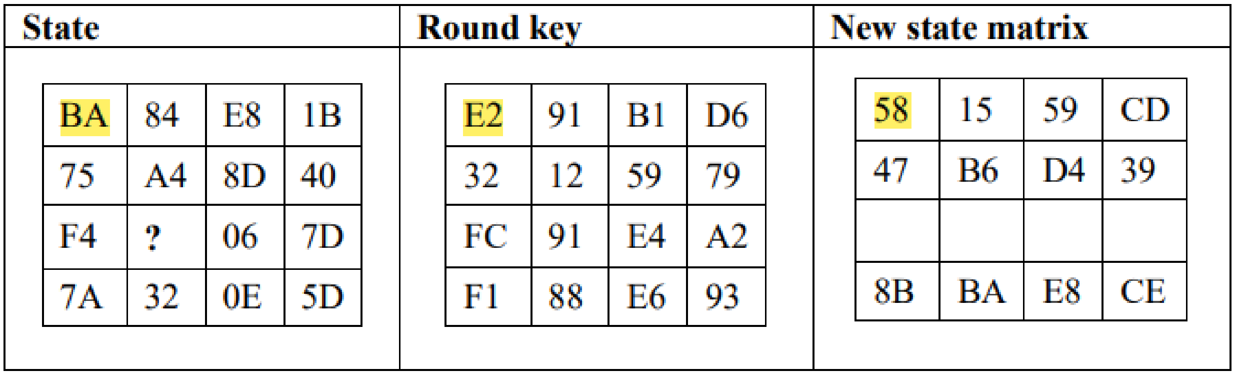
3. **{01 . C7}**
   C7 = 11000111

4. **{01 . 20}**
   20 = 00100000

Answer :
= {02 . EB} + {03 . 93} + {01 . C7} + {01 . 20}
= 11001101 XOR 10101110 XOR 11000111 XOR 00100000
= 10000100 (84 in hex)

Step 4 : Add Round key

Example 1:



State

BA	84	E8	1B
75	A4	8D	40
F4	?	06	7D
7A	32	0E	5D

Round key

E2	91	B1	D6
32	12	59	79
FC	91	E4	A2
F1	88	E6	93

New state matrix

58	15	59	CD
47	B6	D4	39
8B	BA	E8	CE

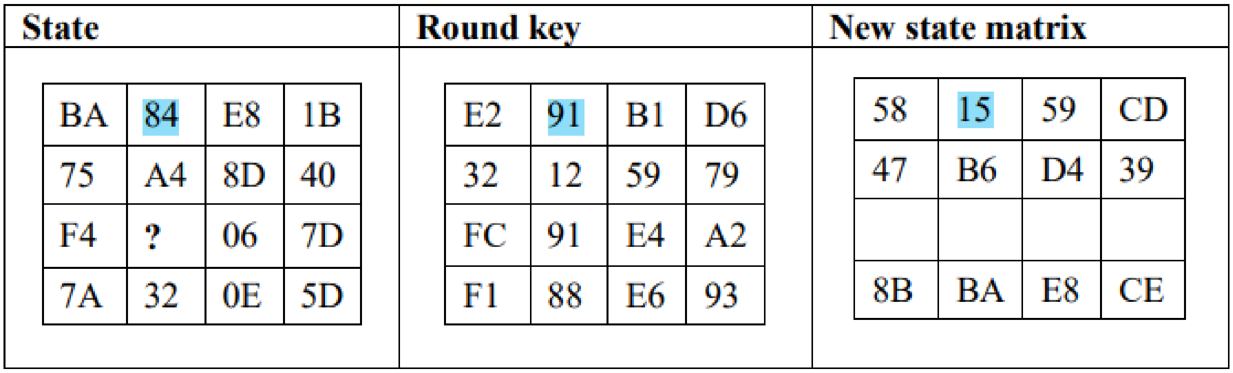
Add round key steps:

1. Convert state (*BA*) into binary format :
   BA = 10111010

2. Convert round key (*E2*) into binary format :
   E2 = 11100010

3. Combine state with the round subkey using the XOR operation (⊕)
   10111010 XOR 11100010 = 1011000 (58)

Example 2:



State

BA	84	E8	1B
75	A4	8D	40
F4	?	06	7D
7A	32	0E	5D

Round key

E2	91	B1	D6
32	12	59	79
FC	91	E4	A2
F1	88	E6	93

New state matrix

58	15	59	CD
47	B6	D4	39
8B	BA	E8	CE

1. State : 84 = 10000100
2. Round key : 91 = 10010001
3. 10000100 ⊕ 10010001 = 00010101 (15 in hex)