I - ASSIGNMENT

(Start Writing From Here)

1. State and prove commutative laws, associative laws and distributive law with Truth table.

Ans: **Commutative Law :-** The Commutative law is applicable only for addition and multiplication operations. But, it is not applied to other two arithematic operations such as subtraction and division. As per commutative law or commutative property, if a and b are any two integers, then the addition and multiplication of a and b result in the same answer even if we change the position of a and b. Symbolically it may be represented as:

$$\boxed{\begin{aligned} a+b &= b+a \\ a\times b &= b\times a \end{aligned}}$$

For example if 2 and 5 are the two numbers, then;

$2+5 = 5+2 = 7$

$2\times 5 = 5\times 2 = 10$.

Definition - The definition of commutative law states that when we add or multiply two numbers then the resultant value remains the same, even if we change the position of the two numbers. Or we can say, the order in which we add or multiply any two real numbers does not change the result.

Using OR Operator $\rightarrow$ $A+B = B+A$.

Using AND Operator $\rightarrow$ $A*B = B*A$.

CMRIT

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

B	A	B + A
0	0	0
0	1	1
1	0	1
1	1	1

$\boxed{\therefore\ A+B = B+A\,.}$

A	B	A * B
0	0	0
0	1	0
1	0	0
1	1	1

B	A	B * A.
0	0	0
0	1	0
1	0	0
1	1	1.

$\boxed{\therefore\ A * B = B * A\,.}$

Associative Law: If a logical operation of any two Boolean variables is performed first and then the same operation is performed with the remaining variable gives the same result, then the logical operation is said to be Associative. The logical OR and logical AND operations of three Boolean variables x, y & z are shown as:-

$$\boxed{\begin{aligned}(x+y)+z &= x+(y+z) \\ (x \cdot y) \cdot z &= x \cdot (y \cdot z)\end{aligned}}$$

Associative Laws obeys for logical OR & logical AND operations.

A	B	C	(A+B)	(A+B)+C	(B+C)	A+(B+C)
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	1	0	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

Hence, from the truth table, we can prove that:-

$$\boxed{\therefore (A+B)+C = A+(B+C).}$$

[P.T.O].

A	B	C	(A·B)	(A·B)·C	(B·C)	A·(B·C)
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

Hence, from the truth table, we have proved that :-

$$\boxed{(A\cdot B)\cdot C = A\cdot(B\cdot C).}$$

Distributive Law :- Distributive property explains that the Operation performed on Numbers, available in brackets that can be distr

X	Y	Z	Y+Z	XY	XZ	X(Y+Z)	XY+XZ.
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	1
1	1	0	1	1	0	1	1
1	1	1	1	1	1	1	1

$$\boxed{\therefore\ X(Y+Z) = (XY)+(XZ).}$$

-buted for each number outside the bracket. It is one of the most frequently used properties in Boolean Algebra.

Defination: The distributive property is an Algebraic property that is used to multiply a single value and two or more values within a set of paranthesis. The distributive property states that when a factor is multiplied by the sum/addition of two terms, it is essential to multiply each of the two numbers by the factor, and finally perform the addition operation. This property can be stated symbollically as:-

$$\boxed{\begin{aligned} A(B+C) &= AB + AC \\ A + (BC) &= (A+B)\cdot(A+C) \end{aligned}}$$

X	Y	Z	YZ	X + YZ	XZ	X+Y	X+Z	(X+Y)(X+Z)
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1	0
0	1	0	0	0	0	1	0	0
0	1	1	1	1	0	1	1	1
1	0	0	0	1	0	1	1	1
1	0	1	0	1	1	1	1	1
1	1	0	0	1	0	1	1	1
1	1	1	1	1	1	1	1	1

$\therefore$

From the Truth table, it is proven that :-

$$\boxed{X + YZ = (X+Y)(X+Z)}$$

2. Explain about floating-point representation with an example.

Ans: A Number with decimal is called as a floating point. The floating-point representation can implement operations for high range values. The Numerical evaluations are carried out using floating-point values. It can create calculations easy. Scientific numbers are described as follows -

→ The Number 5,600,000 can be described as $0.56 * 10^7$.
→ Therefore, 0.56 is the mantissa and 7 is the value of the Exponent.

• Binary numbers can also be described in exponential form. The description of binary numbers in the exponential form is called as "floating-point representation".

• The floating point representation breaks the number into two parts, the left-hand side is signed, fixed-point number known as a mantissa and the right-hand side of the number is known as a exponent.

• The floating-point values are also authorized with a sign; 0 denoting the positive value and 1 denoting the negative value.

(P.T.O).

The General Structure of floating-point representation of a Binary Number –

$$x = (x_0 * 2^0 + x_1 * 2^1 + x_2 * 2^2 \pm \ldots \mp b_{-(n-1)} * 2^{-(n-1)})$$

$\Rightarrow$ $\boxed{\text{mantissa} * 2^{\text{Exponent}}}$.

In the following Syntax, the decimal point is transferred left for negative exponents of two and right for positive exponents of two. Both the mantissa and the exponent are signed values enabling negative numbers and negative exponents commonly.

1. Converting floating point to binary :-

Eg 1 :- $(15.25)_{10} = (?)_2$

$15.25 \longrightarrow 0.25 \times 2 = 0.50 \rightarrow 0$

$(1111.010)_2$ $\quad 0.50 \times 2 = 1.00 \rightarrow 1$

$0.00 \times 2 = 0.00 \rightarrow 0 \quad = \underline{010}$

2. Converting Binary to floating point :-

Eg 1 :- $(1111.01)_2 = (?)_{10}$

$2^3\ 2^2\ 2^1\ 2^0\ \ 2^{-1}\ 2^{-2}$

$\Rightarrow (15 . 0.25) \Rightarrow (15 + 0.25) = \underline{15.25}$

$\Rightarrow$ 1. IEEE Standard 754 : floating point is the most common representation for real numbers in Computers.

$\Rightarrow$ 2. IEEE 754 has 3 basic Components :-

i) Sign of mantissa i.e., 0 for +ve No's. & 1 for -ve No's.

ii) Biased exponent $\rightarrow$ Exponent field needs to represent both the and -ve exponents.

→ A Biased is added to actual exponent in order to get true exponent.

iii) Normalized Mantissa - Actual No's has to be Normalized before storing it in the system.

③ IEEE 754 has 2 different representations to store floating points:-

i) SINGLE PRECISION: Here, length of the word = 32 bits.

1	8	23
Single	exponent	mantissa

32.

Normalization of values - ① $120 = 1.20 \times 10^2$.

② $(153.25)_{10} = 1.5325 \times 10^2$

③ $0.0786 = 7.86 \times 10^{-2}$.

Place decimal value after first digit where, our first digit should not be zero.

Exponent bias - $E' = E + 127$.

Eg 1:- i) $(1010.10101)_2$ Sign: 0

$\Rightarrow 1.01010101 \times 2^3$. exponent: $E' = E + 127$

$= 3 + 127 = \underline{130}$

Normalized mantissa :- 01010101.

ii) DOUBLE PRECISION: Here, the length of the word = 64 bits.

1	11	52
Sign	Exponent	Mantissa

← 64 →

$E' = E + bias$
$E = E + 1023.$

Eg 1 :- $(-10.35)_{10}$

Sign = 1;

i) convert deci to binary no :-

$(-10.35)_{10}$

⇒ −1010.0101

$0.35 \times 2 = 0.70 = 0$
$0.70 \times 2 = 1.40 = 1$
$0.40 \times 2 = 0.80 = 0$
$0.80 \times 2 = 1.60 = 1$

ii) Normalization :-

Mantissa = $1.\underbrace{0100101}_{\text{mantissa}} \times 10^{3}$ → exponent

iii) $E' = E + 1023 = 1026.$

iv) place :- $(10000000001)_{10}.$

3. What do you mean by decoder? Design a 3 to 8 line decoder and explain it.

Ans: A Binary decoder is a digital circuit that converts a binary code into a set of outputs. The Binary code represents the position of the desired output and is used to select the specific output that is active. Binary decoders are the inverse of encoders and are commonly used in digital systems to convert a serial code into a parallel set of outputs.

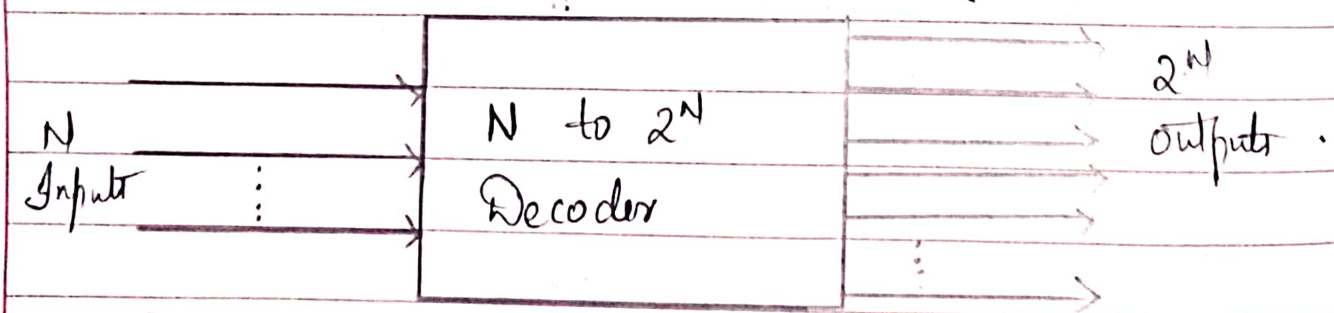
→ It works opposite to encoder. It is a combinational circuit, which converts n lines of input to 2^n lines of output.

1. The Basic principle of a binary decoder is to assign a unique output to each possible binary code. For example, a binary decoder with 4 Inputs and $2^4 = 16$ outputs can assign a unique output to each of the 16 possible set of 4-bit binary codes.

2. The Inputs of a binary decoder are usually active low, meaning that only one Input is active (low) at any given time, and the remaining Inputs are inactive (high). The active low Input is used to select the specific output that is active.

3. There are different types of binary decoders, Including priority decoders, which assign a priority to each output, and error-detecting decoders, which can detect errors in the binary code and generate an error signal.

In Summary, a binary decoder, as the name "Decoder" means to translate or decode coded Information from one format to another, so a digital decoder transforms a set of digital Input signals into an equivalent decimal code at its output.



3 to 8 Decoder and truth table of 3 to 8 decoder :-

- A 3-to 8 decoder has three inputs (A, B, C) and eight outputs (D0 to D7).
- Based on the 3 Inputs one of the eight outputs is selected.
- The truth table for 3-to 8 decoder is shown in the below table.
- From the truth table, it is seen that only one of eight outputs (D0 to D7) is selected based on three select Inputs.
- From the truth table, the logical expressions can be written as :-

A	B	C	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

$D0 = \bar{A}\bar{B}\bar{C}$; $D1 = \bar{A}\bar{B}C$; $D2 = \bar{A}B\bar{C}$; $D3 = \bar{A}BC$; $D4 = A\bar{B}\bar{C}$;
$D5 = A\bar{B}C$; $D6 = AB\bar{C}$; $D7 = ABC$.

- Using the above expressions, the circuit of a 3-to 8 decoder can be implemented using three NOT gates and eight 3-input AND gates.

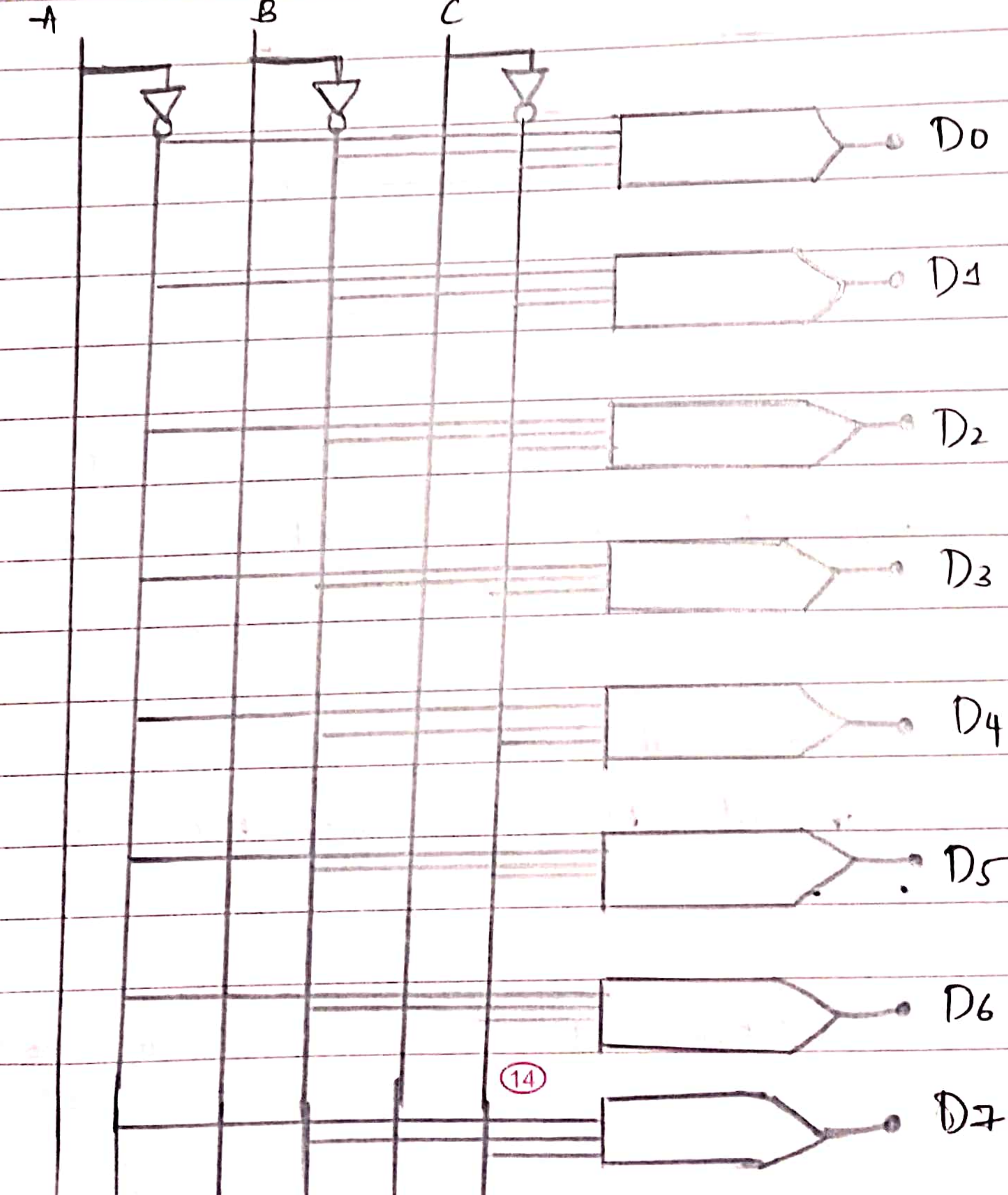
- The three Inputs A, B, and C are decoded into eight outputs, each output representing one of the mid terms of the 3-input variables

• The three Inverters provide the complement of the inputs and each one of the eight AND gates generates one of the midterms.

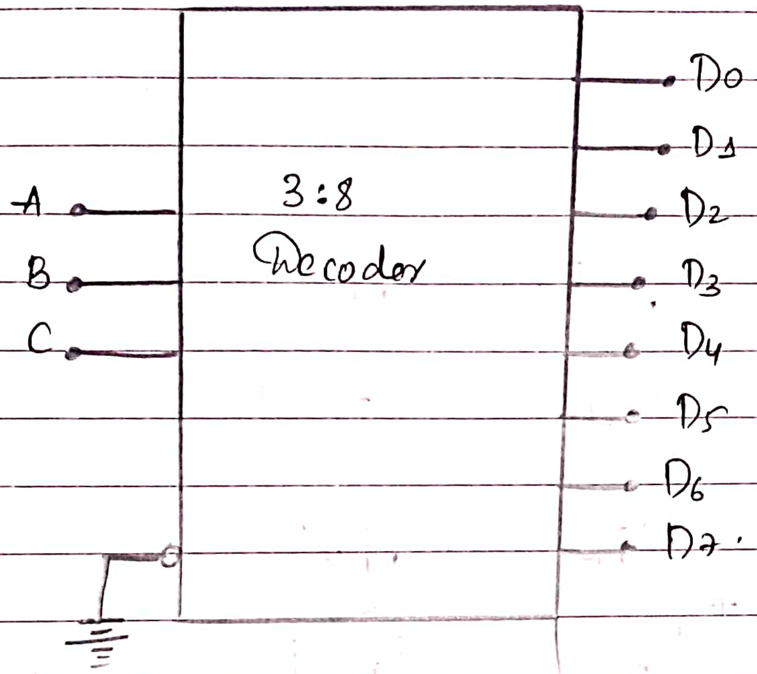
• This decoder can be used for decoding any 3-bit code to provide eight outputs, corresponding to eight different combinations of the input code.

• This is also called as a 1 of 8 decoder since only one of eight output lines is HIGH for a particular input combination.

Logic diagram of 3 to 8 decoder



It is also called as a binary to octal decoder since it takes the inputs representing 3-bit binary numbers and the outputs representing the eight digits in the octal Number System.



3-to 8 decoder in IC form

4. How does a J-K flipflop differs from an S-R flipflop in its basic operations? Explain.

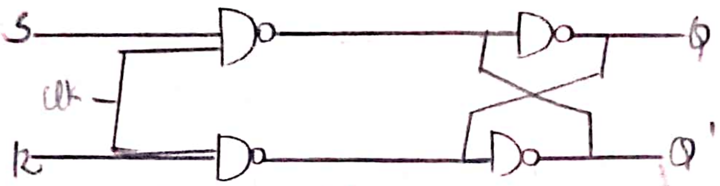
Ans: SR Flipflop:- Operates only with positive and Negative clock transi-tion. It takes 2 Inputs S(set) and R(reset). The two outputs Q and Q' can be constructed in two ways:-

① Two NOR gates + Two AND gates

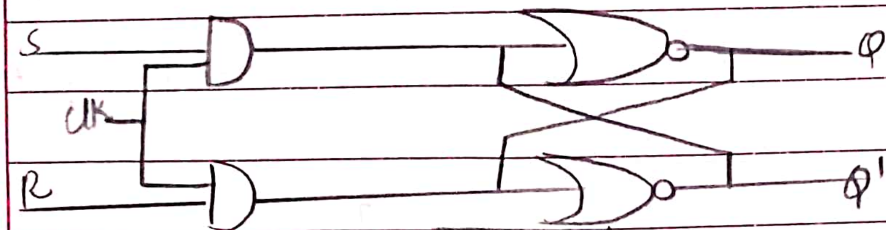
② four NAND gates.

⇒ ① Circuit diagram with two NOR and two AND gates :-

CMRIT (D-7-0)

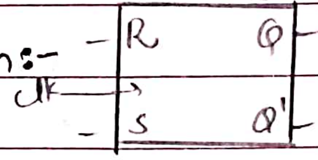


} for four NAND gates.

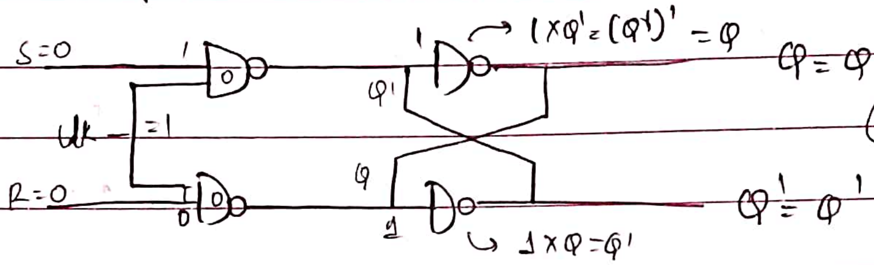


} for two NOR + two AND gates

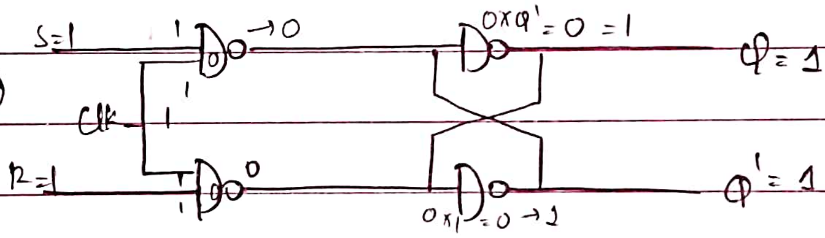
Block Diagram:-



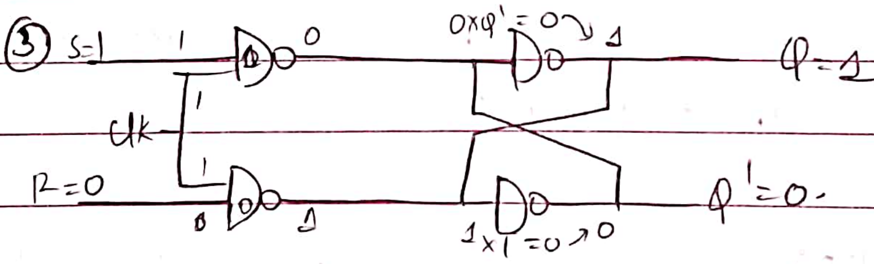
SRFF :-

① 

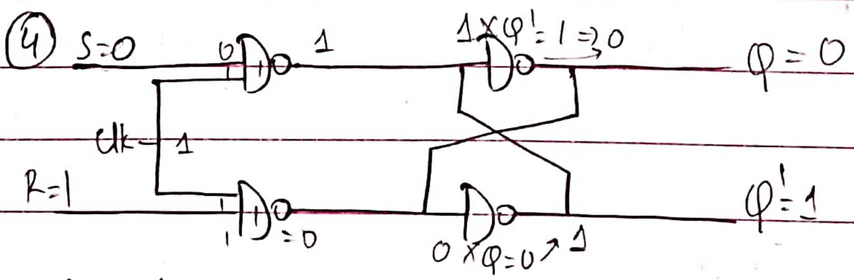
(No change)

② 

(Invalid)

③ 

(1,0).

④ 

(0,1).

Truth table :-

Clk	S	R	Q	Q'
0	X	X	Hold.	
1	0	0	0	1
1	0	1		
1	1	0	1	0
1	1	1	Invalid	

Characteristic Table:-

	S	R	Q(n)	Q(n+1)
0	0	0	0	0 } hold.
1	0	0	1	1
2	0	1	0	0 } Reset
3	0	1	1	0
4	1	0	0	1 } set
5	1	0	1	1
6	1	1	0	X } Invalid.
7	1	1	1	X

Characteristic Equation:-

S \ RQ(n)	00	01	11	10
0	0	1 (1)	3	2
1	1 (4)	1 (5)	X (7)	X (6)

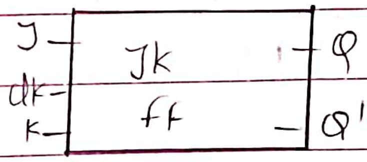
$\Rightarrow S + R'Q_n$

Excitation Table:-

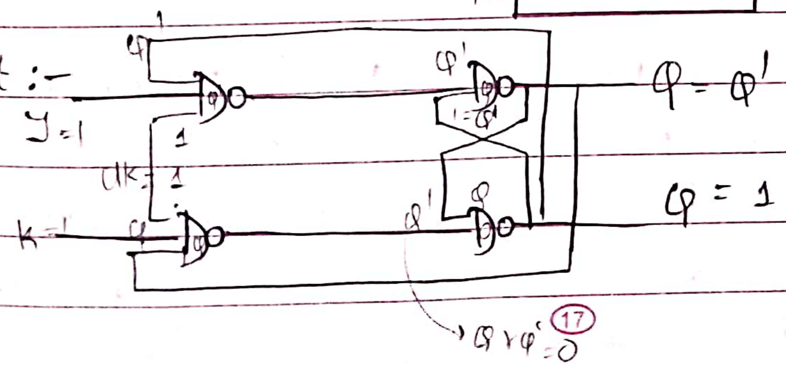
Q_n (Q)	Q_{n+1} (Q')	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0.

JK flipflop:- It is a modified version of SR flipflop. It Operates only with +ve and -ve clock transition. It has two Inputs J & K. And two outputs Q, Q'. It is widely used flipflop. It has two levels of Latches, one acts as Master and Second acts as Slave. Always master responds first, next Slave.

JKFF Logical Symbol:-



Circuit:-



Characteristic Table :-

CLK	J	K	Q	
0	X	X	No change	
1	0	0	No change.	
1	0	1	0	1
1	1	0	1	0
1	1	1	toggle	

Characteristic Equation :-

J \ KQn	00	01	11	10
0	(0)	1 (1)	(3)	(2)
1	1 (4)	1 (5)	(7)	1 (6)

$\Rightarrow Q_{n+1} = JQ' + K'Q.$

Excitation Table :-

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0.

5. Explain about Basic Computer Register.

Ans :- Registers are a type of computer memory used to quickly accept, store and transfer data and Instructions that are being used Immediately by the CPU. The registers used by CPU are often termed as Processor registers.

- A Processor register may hold an Instruction, a storage address, or any data (such as bit sequence or Individual characters).

The computer needs processor registers for manipulating data and a register for holding a memory address. The register holding the memory location is used to calculate the address of the next instruction after the execution of the current instruction is completed.

Register	Symbol	Number of bits	Function.
Data register	DR	16	Holds memory operand
Address register	AR	12	Hold address for the memory.
Accumulator	AC	16	Process register.
Instruction register	IR	16	Holds instruction code.
Program counter	PC	12	Holds address of the instruction
Temporary register	TR	16	Holds temporary data
Input register	INPR	8	Carries input character
Output register.	OUTR	8	Carries output character.

- The Memory unit has a capacity of 4096 words, and each word contains 16 bits.
- The Data Register (DR) contains 16 bits which hold the operand read from the memory location.
- The Memory Address Register (MAR) contains 12 bits which hold the address for the memory location.
- The Program Counter (PC) also contains 12 bits which hold the address of the next instruction to be read from memory after the current instruction is executed.
- The Accumulator (AC) register is a general purpose processing register.
- The instruction read from memory is placed in the instruction register (IR)
- The Temporary Register (TR) is used for holding the temporary data during the processing.
- The Input Register (IR) holds the input characters given by the user.
- The Output Registers (OR) holds the output after processing the input data.