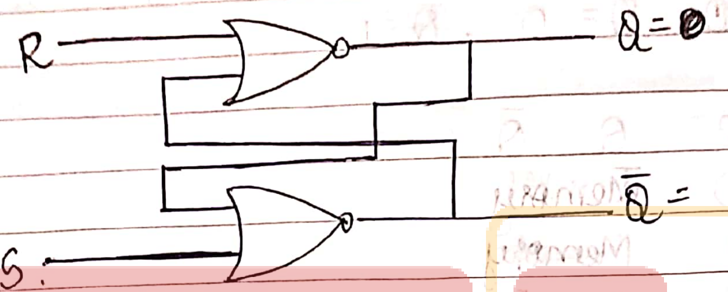
30/9/19

MODULE : 04

LATCHES

Latches: It is a basic storage of element & it has no control on any of the inputs.

* NOR truth table for SR latch :



NOR

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

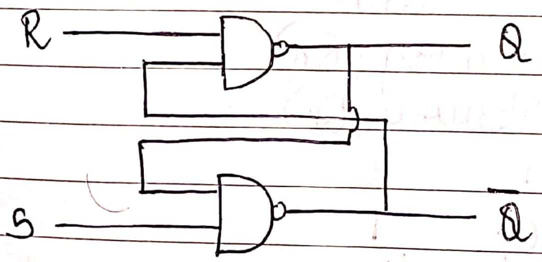
Case I: $S=0,\ R=1,\ Q=0,\ \bar{Q}=1$
$S=0,\ R=0,\ Q=0,\ \bar{Q}=1$ (Memory).

Case II: $S=1,\ R=0,\ Q=1,\ \bar{Q}=0$.
$S=0,\ R=0,\ Q=1,\ \bar{Q}=0$ (Memory).

Case III: $S=1,\ R=1,\ Q=0,\ \bar{Q}=0$ (Not used / Invalid)
$S=0,\ R=0,\ Q=0,\ \bar{Q}=1$

S	R	Q	$\bar{Q}$
0	0	Memory	
0	1	0	1
1	0	1	0
1	1	Not used	

* NAND truth table for SR latch :



NAND.

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Q_n : Present / Previous state
Q_{n+1} : Next state

Case I: $S=0,\ R=1,\ Q=0,\ \bar{Q}=1$ (Memory)
$S=1,\ R=1,\ Q=0,\ \bar{Q}=1$ ~~(Memory)~~

Case II: $S=1,\ R=0,\ Q=1,\ \bar{Q}=0$ (Memory)
$S=1,\ R=1,\ Q=0,\ \bar{Q}=1$

Case III: $S=0,\ R=0,\ Q=1,\ \bar{Q}=1$ (Not used)
$S=1,\ R=1,\ Q=0,\ \bar{Q}=1$

S	R	Q	$\bar{Q}$
0	0	Invalid	
0	1	0	1
1	0	1	0
1	1	Memory.	

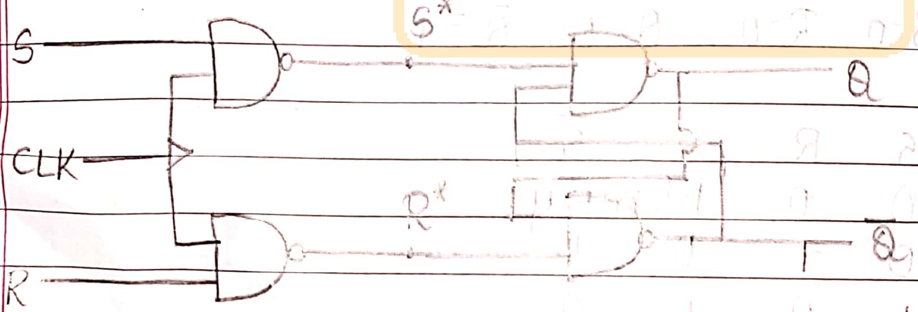
Reference T

S	R	Q	$\bar{Q}$
0	0	Invalid	
0	1	0	1
1	0	1	0
1	1	Memory	

-14m-
V. Imp

SR Flip Flop.

SR Flip Flop using NAND.



$S^* = (\overline{S\cdot CLK}) = \bar{S} + \overline{CLK}$.
$R^* = (\overline{R\cdot CLK}) = \bar{R} + \overline{CLK}$.

TT:

CLK	S	R	Q_n	Q_{n+1}
0	X	X	Memory (Q_n)	
1	0	0	Memory (Q_n)	
1	0	1	1	0
1	1	0	0	1
1	1	1	Invalid.	

CT: Characteristic Table
ET: Excitation Table
STD: State Diagram

Tracing:

$S = 1+0 = 1$
$R = 1+0 = 1$

$S = 1+0 = 1$
$R = 0+0 = 0$

$S = 0+0 = 0$
$R = 1+0 = 1$

$S = 0+0 = 0$
$R = 0+0 = 0$

CT

Q_n	S	R	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

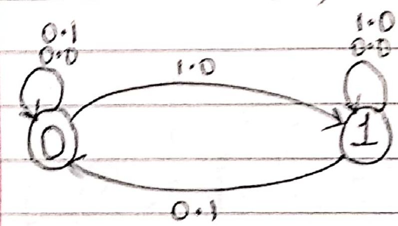
ET

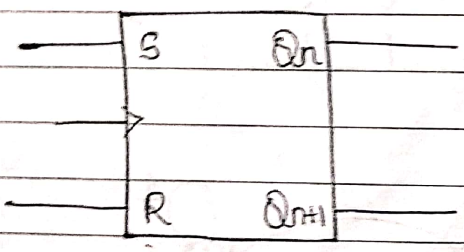
Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Q_n \ SR	00	01	11	10
0	0_0	0_1	X_3	1_2
1	1_4	0_5	X_7	1_6

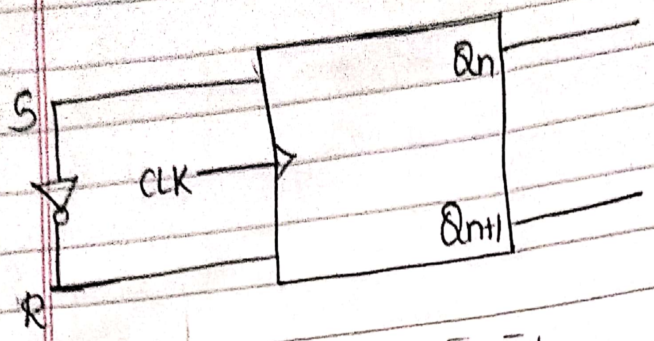
$$Q_{n+1} = S + Q_n\overline{R}$$

STD $(Q_n \to Q_{n+1})$



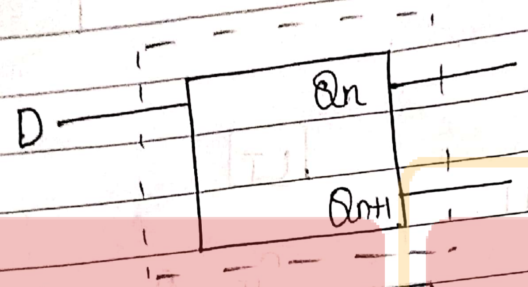


D - Flip Flop - (Data flip flop)



Reference TT

CLK	S	R	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	Inval



TT

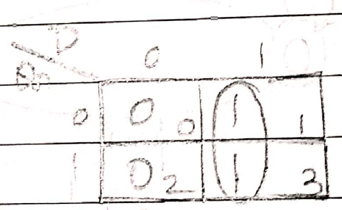
CLK	D	Q_{n+1}
0	X	Q_n
1	0	0
1	1	1

CT

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

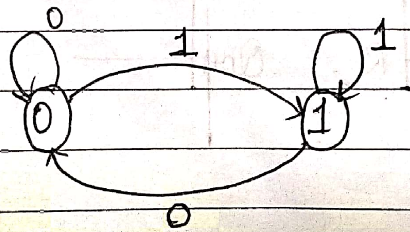
ET

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1



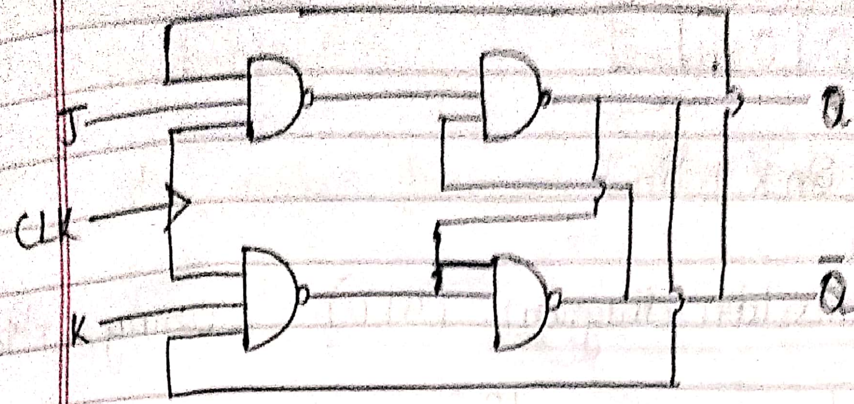
$Q_{n+1} = D$

STD



04/10

JK Flip Flop



CLK	S	R	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	Invali

CLK	J	K	Q_{n+1}
0	X	X	Q_n } Memory
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	$\bar{Q}_n$ (Toggle)

Case 4 : $J=1, K=1, Q=?, \bar{Q}=$

Assume $Q=0, \bar{Q}=1$.

$Q=1, \bar{Q}=0$.

CT Table:

(Using JK T.T)

	i/p		o/p
Q_n	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

ET:

(Using CT table)

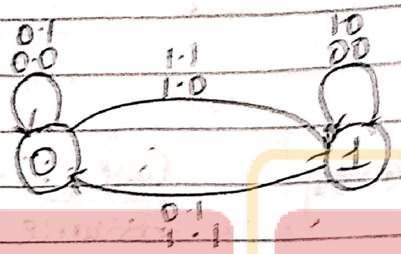
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Q_n \ JK	00	01	11	10
0	0	0	1	1
1	1	0	0	1

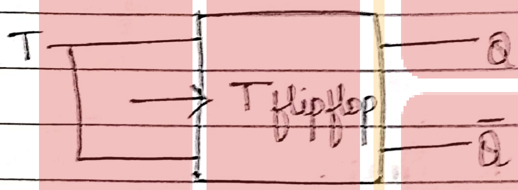
(Using CT table)

$$Q_{n+1} = Q_n \bar{K} + \bar{Q}_n J$$

State Transition diagram (STD) (using CT table)



T- Flip Flop



CLK	J	K	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	$\bar{Q}_n$ (Toggle)

CLK	T	Q_{n+1}
0	X	Q_n
1	0	Q_n
1	1	$\bar{Q}_n$

CT table:

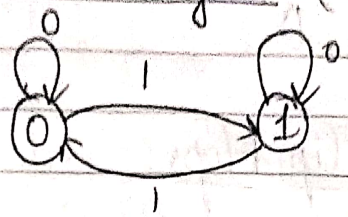
Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

ET

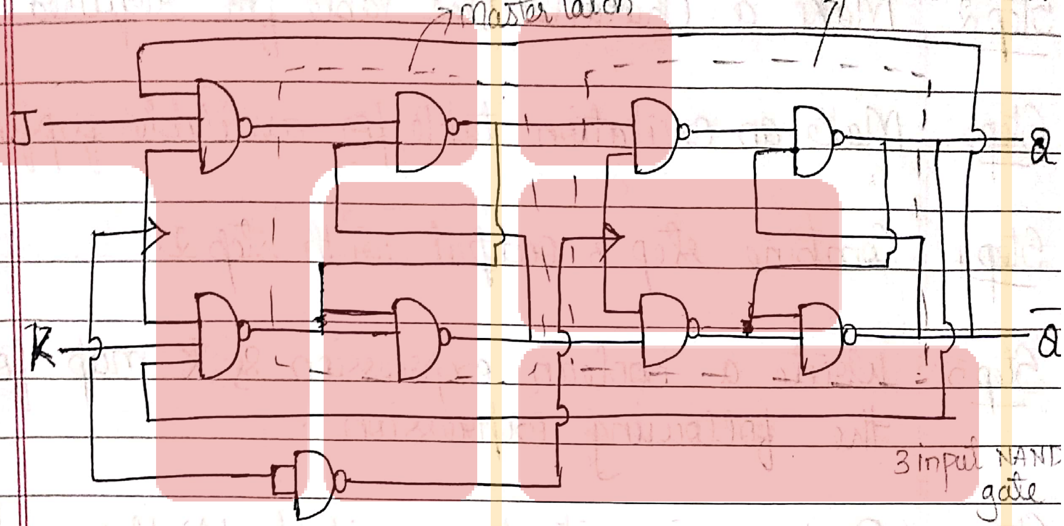
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Q_n \ T	0	1
0	0_0	1_1
1	1_2	0_3

State Transition diagram: (STD)



JK - Masterslave flip flop.



CLK	J	K	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	$\bar{Q}_n$ (Toggle)

CT table:

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

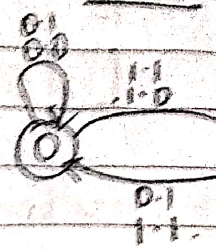
Q_n \ JK	00	01	11	10
0	0_0	0_1	1_3	1_2
1	1_4	0_5	0_7	1_6

$$Q_{n+1} = Q_n\bar{K} + \bar{Q}_n J$$

ET:

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

STD:



10/10/19

Conversion of Flip Flops.

Step 1: Identify the available and required flipflop.

Step 2: Make a Characteristic table for required flipflop

Step 3: Make an excitation table for available flipflop.

Step 4: Combine step 3 output with step 2.

Step 5: Write a boolean expression & K-map for the following expression.

Step 6: Design a circuit for available flipflop and required flipflop

(1) Convert JK flipflop into D flip flop.

Soln: Step I: Available FF - JK flipflop.
Required FF - D flipflop.

Step II: CT for D.

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Step III: ET for JK.

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

Step IV:

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

Step V:

J =

Q_n \ D	0	1
0	0_0	1_1
1	X_2	X_3

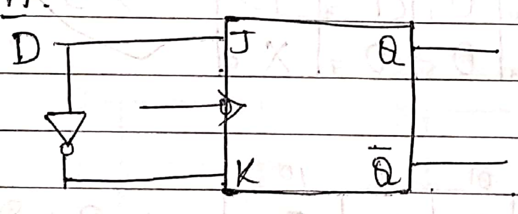
$\Rightarrow J = D$.

K =

Q_n \ D	0	1
0	X_0	X_1
1	1_2	0_3

$\Rightarrow K = \overline{D}$

Step VI:



② Convert SR flipflop into JK flipflop.

Soln:

Step 1: Available FF - SR flip flop.
Required FF - JK flip flop.

Step 2:

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

Step 3

Q_n	Q_{n+1}	S
0	0	0
0	1	1
1	0	0
1	1	X

Step 4:

Q_n	J	K	Q_{n+1}	S	R
0	0	0	0	0	X
0	0	1	0	0	X
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	X	0
1	0	1	0	0	1
1	1	0	1	X	0
1	1	1	0	0	1

Step 5:

S =

Q_n \ JK	00	01	11	10
0	0	0	1	1
1	X	0	0	X

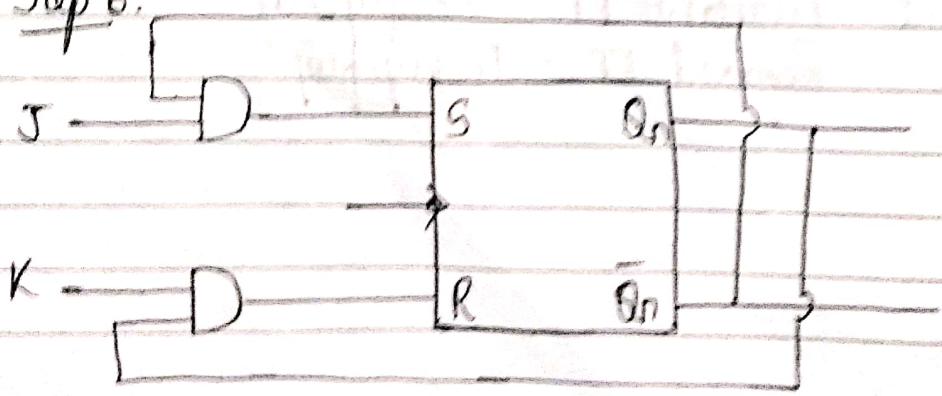
$\Rightarrow S = \bar{Q}_n J$

R =

Q_n \ JK	00	01	11	10
0	X	X	0	0
1	0	1	1	0

$\Rightarrow R = Q_n K$

Step 6:



(3) Convert SR flipflop to D flipflop.

Soln: Step 1: Available FF - SR flipflop
Required FF - D flipflop.

Step 2:

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Step 3:

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Step 4:

Q_n	D	Q_{n+1}	S	R
0	0	0	0	X
0	1	1	1	0
1	0	0	0	1
1	1	1	X	0

Step 5:

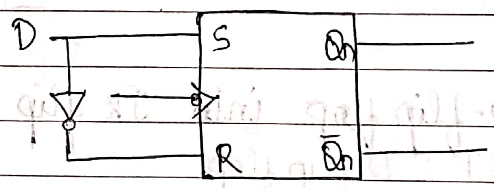
Q_n \ D	0	1
0	0_0	1_1
1	0_2	X_3

$S = D$.

Q_n \ D	0	1
0	X_0	0_1
1	1_2	0_3

$R = \overline{D}$.

Step 6:



(4) Convert SR flipflop to T flipflop.

Soln: Step 1: Available FF - SR flip flop.
Required FF - T flipflop.

Step 2:

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Step 3:

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Step 4:

Q_n	T	Q_{n+1}	S	R
0	0	0	0	X
0	1	1	1	0
1	0	1	X	0
1	1	0	0	1

Step 5:

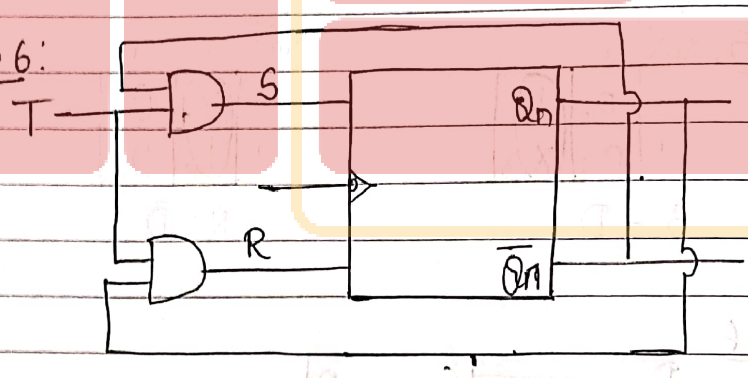
Q_n \ T	0	1
0	0	1
1	X	0

$S = \overline{Q_n}T$

Q_n \ T	0	1
0	X	0
1	0	1

$R = Q_nT$

Step 6:



12/10/19

(5) Convert D-flip flop into JK flip flop.

Soln: * Available FF : D flip flop.
Required FF : JK flip flop.

* Characteristic table for JK.

Q_n	J	K	Q_{n+1}
0	0	0	~~Q_n~~ 0
0	0	1	~~Q_n~~ 0
0	1	0	~~0~~ 1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0

* ET for D-FF.

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

*

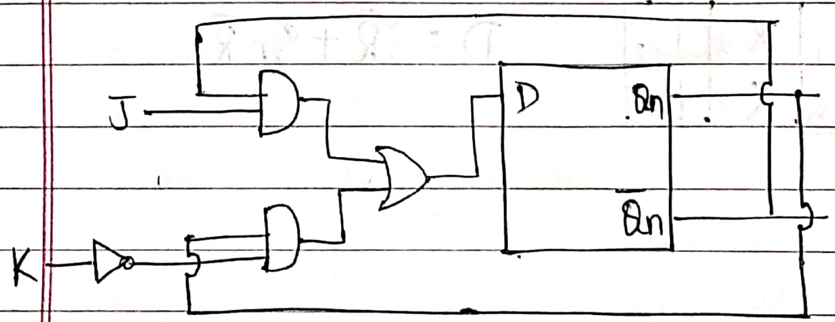
Q_n	J	K	Q_{n+1}	D
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

*

D ⇒

Q_n \ JK	00	01	11	10
0	0_0	0_1	1_3	1_2
1	1_4	0_5	0_7	1_6

$$D = \overline{Q_n} J + Q_n \overline{K}.$$



(6) Convert D-flip flop to SR flip flop.

Soln: Available - D·FF

Required - SR·FF.

CT for SR

Q_n	S	R	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

ET for D

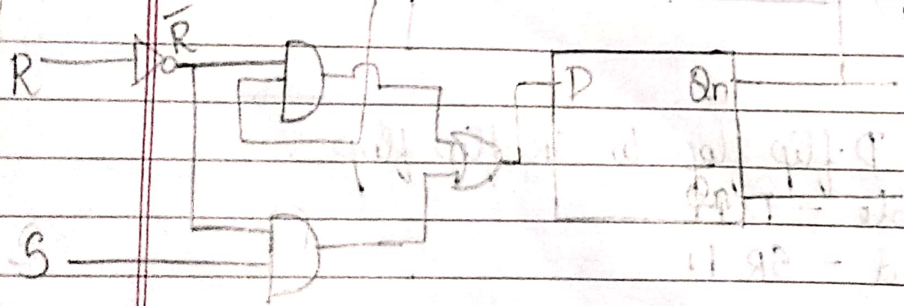
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Q_n	S	R	Q_{n+1}	D
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	X	X
1	0	0	1	1
1	0	1	0	0
1	1	0	1	1
1	1	1	X	X

D:

Q_n \ SR	00	01	11	10
0	0_0	0_1	X_3	1_2
1	1_4	0_5	X_7	1_6

$D = S\bar{R} + Q_n\bar{R}$



(7) Convert D flipflop to T flip flop.

Soln: Available : D FF.
Required : T FF.

ET for D.

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

CT for T.

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Q_n	T	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

Q_n \ T	0	1
T = 0	0_0	1_1
1	1_2	1_3

~~D~~ T = $\overline{Q_n}$

(8) Convert JK flip flop to SR flip flop.

Soln: Available : JK flip flop.
Required : SR flip flop.

CT for SR.

Q_n	S	R	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

ET for JK.

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

Q_n	S	R	Q_{n+1}	J	K
0	0	0	0	0	X
0	0	1	0	0	X
0	1	0	1	1	X
0	1	1	X	X	X
1	0	0	1	X	0
1	0	1	0	X	1
1	1	0	1	X	0
1	1	1	X	X	X

J =

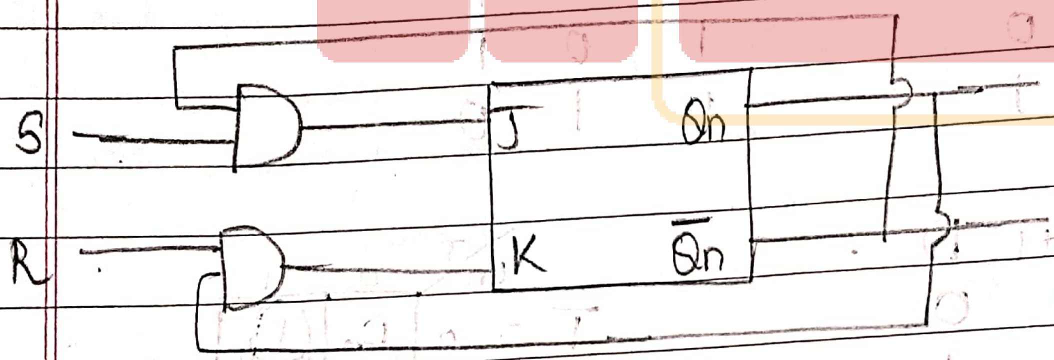
Q_n \ SR	00	01	11	10
0	0_0	0_1	X_3	1_2
1	X_4	X_5	X_7	X_6

$$J = \overline{Q_n} S$$

K =

Q_n \ SR	00	01	11	10
0	X_0	X_1	X_3	X_2
1	0_4	1_5	X_7	0_6

$$K = Q_n R$$



01/08/19.

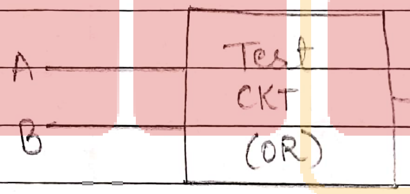
HDL

Format of HDL.

HDL – Hardware discription language.
VHDL – Very high speed integrated circuit hardware description language.

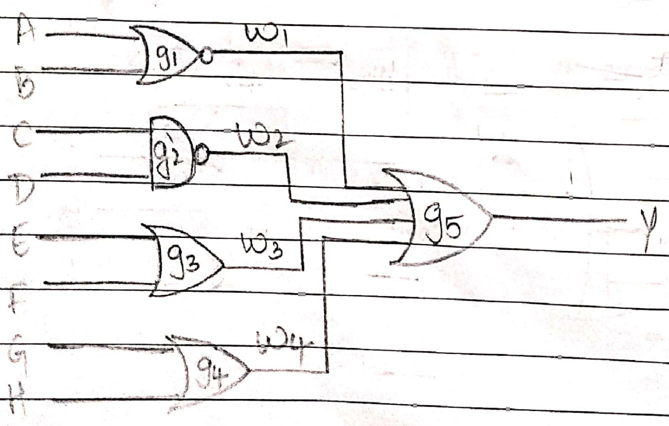
Module Name of the module (parameter used);
input variable;
Output variable;
Body of the program.
// Logic //.
End module;

Test Circuit.



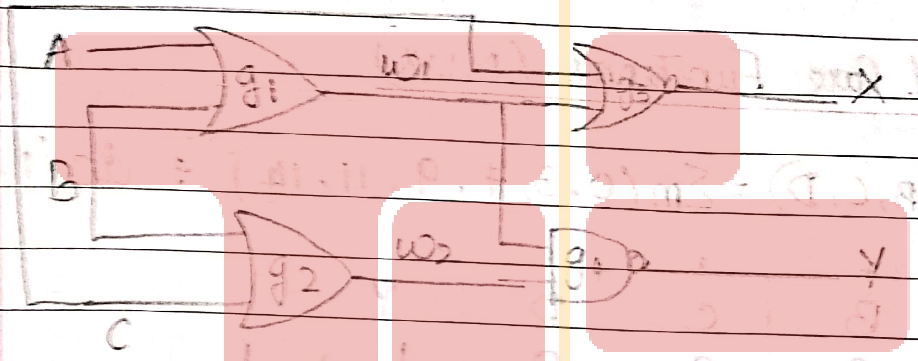
Module Test CKT (A,B,Y);
input A, B;
output Y;
OR g1 (Y, A, B);
End module;

1.



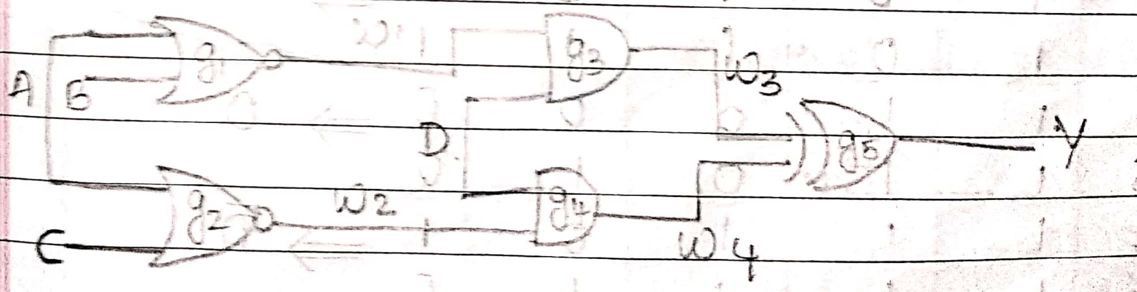
```
Module CKT1 (A,B,C,D,E,F,G,H,Y);
input A,B,C,D,E,F,G,H;
Output Y;
wire w1,w2, w3, w4; w5,
NOR g1(w1, A, B);
NAND g2(w2, C, D);
OR g3(w3, E, F);
OR g4(w4, G, H);
NOR g5(Y, w1, w2, w3, w4);
End module;
```

2.




```
Module CKT2 (A,B,C, X,Y);
input A,B,C;
output X,Y;
wire w1, w2;
OR g1(w1, A,B);
OR g2(w2, B,C);
NOR g3(X, C, w1);
NAND g4(Y, w1,w2);
End module;
```

3.



```
Module CKT2 (A, B, C, D, Y);
input A, B, C, D;
Output Y;
wire w1, w2, w3, w4;
NOR g1 (w1, A, B);
NOR g2 (w2, A, C);
AND g3 (w1, w3, D);
AND g4 (w2, w4, D);
XOR g5 (Y, w3, w4);
End module;
```