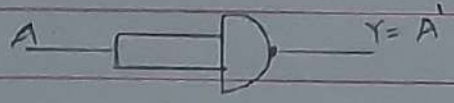
I - ASSIGNMENT

(Start Writing From Here)

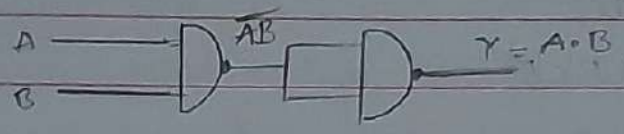
1 How to implement NOT, AND and OR gate using NAND and NOR gates.

1) Implementing AND, OR, NOT gates using Nand

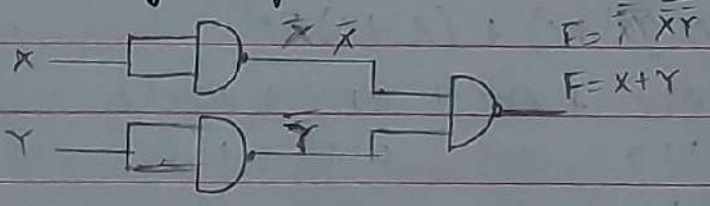
i) NOT by using NAND



ii) AND by using NAND

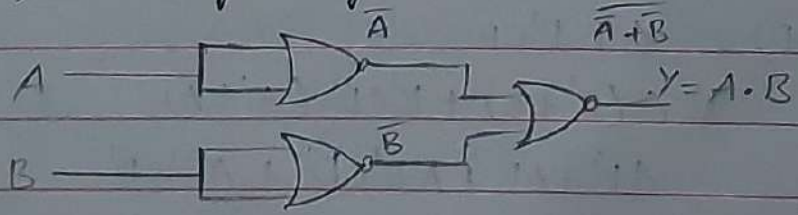


iii) OR by using NAND



2) Implementing AND, OR, NOT by using NOR gate

i) AND by using NOR



ii) OR gate by using NOR

iii) NOT gate by using NOR

2. Find out sop for the following

A) $F(A,B,C) = AB + BC'$

B) $F(A,B,C,D) = A + B'CD'$

A) $F(A,B,C) = AB + BC'$

$AB(C'+C) + (A'+A)BC'$ $\quad \therefore \{C'+C=1, A'+A=1\}$

$ABC' + ABC + A'BC' + ABC'$

1 1 0 $\quad$ 1 1 1 $\quad$ 0 1 0 $\quad$ 1 1 0

$m_6 \quad m_7 \quad m_2 \quad m_6$

∴ This is sop form of given equation.

B) $F(A,B,C,D) = A + B'CD'$

$A(B'+B) + B'CD'(A'+A)$

$AB' + AB + A'B'CD' + AB'CD'$

$AB'(C'+C) + AB(C'+C) + A'B'CD' + AB'CD'$

$= AB'C' + AB'C + ABC' + ABC' + A'B'C'D' + AB'C'D'$

$= AB'C'(D'+D) + AB'C(D'+D) + ABC'(D'+D) + ABC(D'+D)$
$+ A'B'C'D' + AB'C'D'$

$= AB'C'D' + AB'C'D + AB'CD' + AB'CD + ABC'D' + ABC'D$
$+ ABCD' + ABCD + A'B'C'D' + AB'C'D'$ //

∴ This is SOP form for given $F(A,B,C,D)$

3. Demonstrate the design steps of synchronous counters with suitable examples?

Design of synchronous counter

A synchronous counter is a type of counter in which all the flip flops are triggered simultaneously by the same clock pulse

step1 :- Determine the number of flip-flops required.

Firstly, analyse the problem description and determine the number of flip-flops required to implement the synchronous counter. If n is the required number of flip flops, then the smallest value of n is such that the number of states N is less than or equal to 2^n.

step 2:- Draw the state diagram.

Secondly, draw the state diagram that shows all the possible states. The states diagram is basically a graphical way of representing the sequence of states through which the counter progress. In the state diagram, we can also include the case in which the counter goes to a particular state from the invalid states on the next clock pulse.

step 3:- selection of flip-flops and excitation table.

In the third step, select a particular type of flip-flop to be used to implement the counter and draw its excitation table. The excitation table is one that gives the information about the present states, next states, and required excitations of the flip-flop.

step 4:- obtain the minized expression for excitation

Now, obtain the minimal expressions for the excitations of the flipflops by using any minization technique such as K-map.

step 5:- Draw the logic diagram

Draw the logic circuit diagram according to the minimal expression obtained in the step 4.

Example:

Design a synchronous counter using D flip flops that goes through states 0, 1, 2, 4, 0. The unused states must always go to zero on the next clock pulse

solution:-

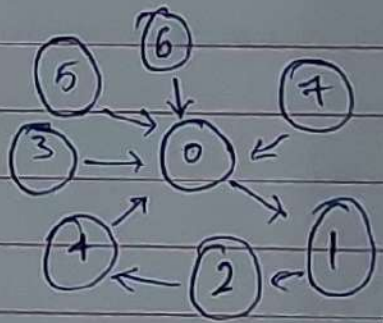
step 1:- Number of flip flops required.

This synchronous counter has for stable states, i.e., 0(000), 1(100), 2(00), 4(100). But we require three flip flops because it counts 4(100) as well.

Since three flip-flops can count eight states. Thus, the remaining four states, i.e. 3(011), 5(101), 6(110) and 7(111) are unused states, the unused states must go to 0(000) after the next clock pulse

step 2:- Draw the state diagram:

The state diagram of the 0, 1, 2, 4, 0... counter



state diagram

step 3:- choose the type of flip flop and write the excitation table

Present state			Next state			Required Excitation		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	D_3	D_2	D_1
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0
0	1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0

step 4:- Derive the minimal expression

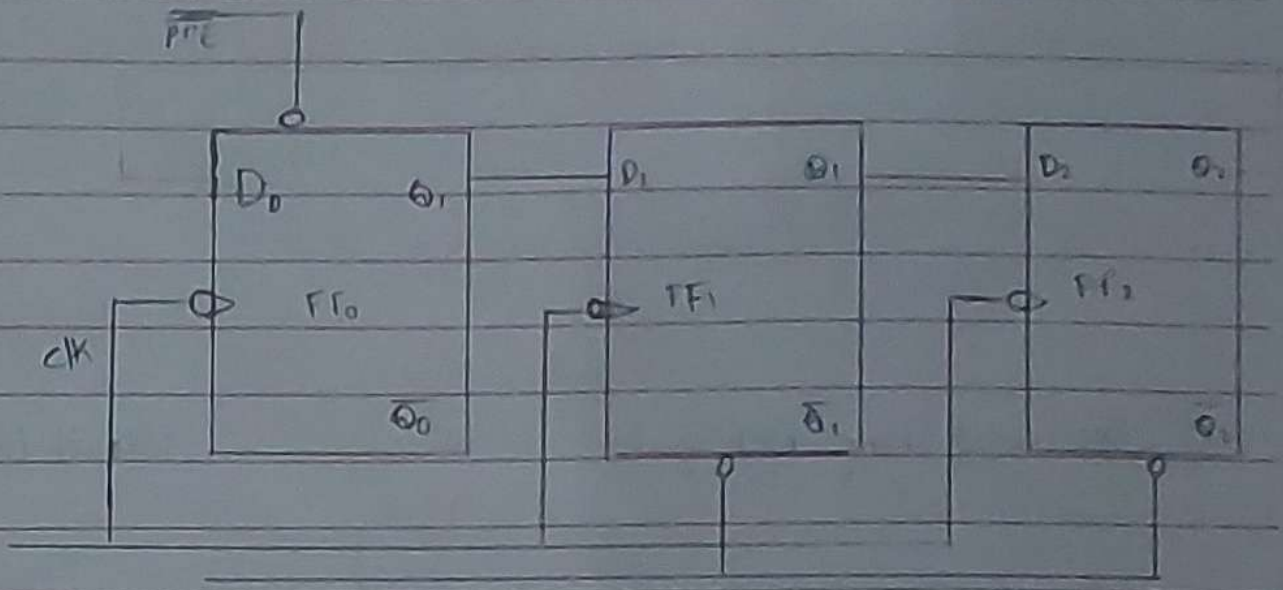
From the excitation table, we can see that there is no minization is possible. Hence, the expressions for the excitations can be directly written from the excitation table itself as follows -

$D_1 = Q_3' Q_2' Q_1'$

$D_2 = Q_3' Q_2' Q_1'$

$D_3 = Q_3' Q_2' Q_1$

step 5:- Draw the logic circuit diagram



Logic circuit Diagram of Counter

4) Explain about S-R latch with a neat diagram using NAND and NOR gates

AN → S-R Latch :-

They are also known as present and clear states. The S-R latch forms the basic building blocks of all other types of flip-flops

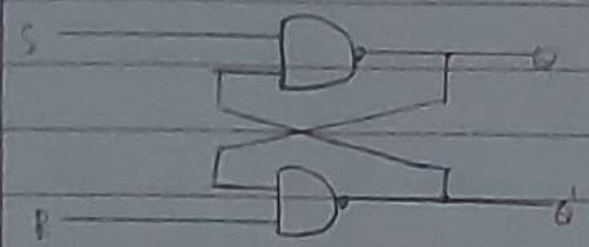
SR latch in a circuit with

i) 2 cross-coupled NOR gate or 2 cross-coupled NAND gate

ii) 2 input s for SET and R for ~~R~~ RESET

iii) 2 output Q, Q'

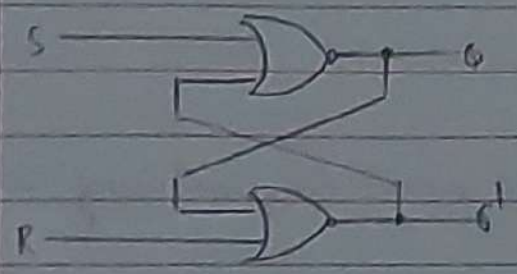
i) S-R Latch using NAND gates



ii) truth table

S	R	Q	Q'
0	0	Invalid condition (not used)	
0	1	1	0
1	0	0	1
1	1	memory / hold state	

i) SR - latch using NOR gates



ii) truth table

S	R	Q	Q'
0	0	memory / hold state	
0	1	0	1
1	0	1	0
1	1	Invalid state (not used)	

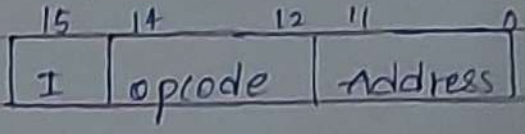
5) what are the different types of Instruction codes?

Types of Instruction code.

In general, there are three codes formats for computer instructions of 16-bit code, the opcode usually contains 3 bits and the rest of 13-bit is subjective, depending on the operation code encountered.

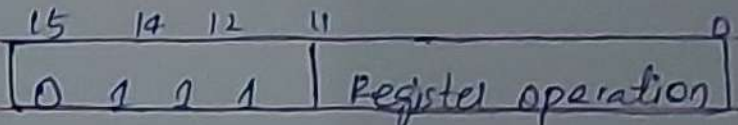
1) memory-reference Instruction :-

In this type of code, 12 bits are used to specify the operation memory address, 3 bits for the opcode (000 to 110), and 1 bit to specify the mode as indirect addressing mode (1)



2) Register-reference instruction :-

12 bits indicate the register operation address, 3 bits for the opcode (111), and 1 bit is utilised for setting the mode as 0, the instructions are executed on the register



3) Input-output instruction :-

This type of code contains a 12-bit input/output operation address, 3 bits for the opcode (111) and 1 bit is utilised for setting the mode as 1.

These instructions are required to transfer to and from the AC register and output device