Module-3

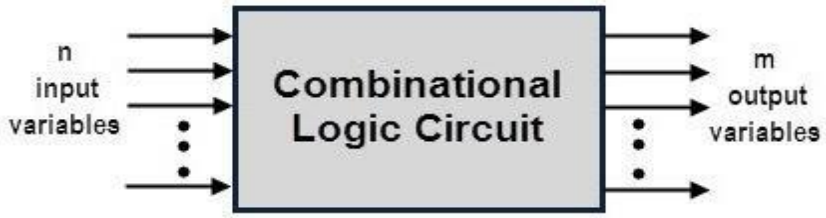
Combinational Circuit Design & Simulation Using Gates

Contents:

- **Review of combinational circuit Design**
- **Design of circuits with limited Gate Fan-in**
- **Gate Delays and Timing Diagrams**
- **Hazards in combinational Logic**
- **Simulation and testing of logic circuits**
- **Multiplexers**
- **Three state buffers**
- **Decoders**
- **Encoders**
- **Programmable Logic Devices**
- **Programmable Logic Arrays**
- **Programmable Array Logic**

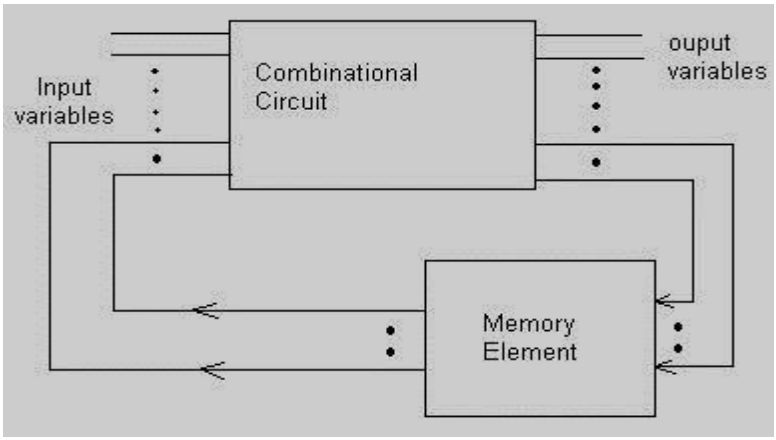
Combinational Circuits:

A combinational circuit is defined as a circuit in which the outputs are dependent on only present inputs.



Sequential Circuits:

A sequential circuit is defined as circuit in which outputs depend on present inputs as well as past outputs.



Differences Between combinational and Sequential circuits

Combinational circuits	Sequential Circuits
➢ In combinational circuits, outputs are dependent on only inputs ➢ Memory element is not required ➢ Combinational circuits are faster ➢ Combinational are easy to design Eg. Decoder	➢ In sequential circuits, the outputs are dependent on present inputs and past outputs. ➢ Memory elements are required ➢ Sequential circuits are slower ➢ Sequential circuits are difficult to design E.g. Counters

Review of Combinational Circuit Design:

- The first step in design of combinational switching circuit is usually to set up a truth table which specifies the output as a function of input variables.
- For **n input variables** the table will have **2^n rows.**
- If any input combination never occur, the corresponding output value is **don't care.**
- The next step is to derive simplified algebraic expression using Karnaugh map, Quine-McCLuskey method.
- The simplified algebraic expression is used to design logic circuit.
- The minimum SOP or POS equation leads directly to a minimum two-level gate circuit.
- **Design of multi-level, multiple output NAND gate circuits** are easily accomplished by first designing a circuit of AND and OR gates. If this circuit has an OR gate at each output and is arranged so that an AND gate (or OR-gate) output is never connected to the same type of gate, a direct conversion to a NAND-gate circuit is possible. Conversion is accomplished by replacing all of the AND gates and OR gates with NAND gates.
- **Similarly, design of multi-level, multi-output NOR gates circuit** is most easily accomplished by first designing a circuit of AND and OR gates, in this best starting point is usually the minimum sum-of products expressions for the complements of the output function, after factoring these expressions to the desired form, they are then complemented to get expressions for the output functions, and the corresponding circuit of AND and OR gates is drawn, if this circuit has an AND gate at each output, and an AND gate (or OR gate) output is never connected to the same type of gate, a direct conversion to a NOR gate circuit is possible.

Design of circuits with limited gate Fan-in:

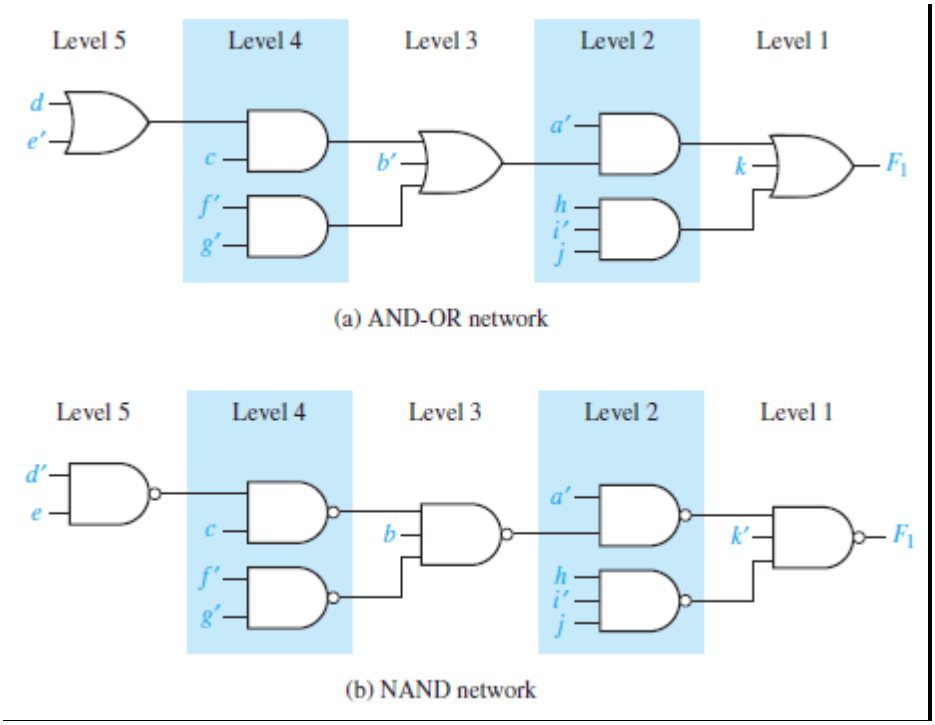
Fan-in: the number of inputs a logic gate can handle.

In practical circuits, the maximum number of inputs on each gate is limited, depending on the type of gates used, this limit may be two, three four, eight or some other number, if a two-level circuit requires more gate inputs than allowed, factoring the logic expression to obtain a multi-level realization is necessary.

Procedure to design of Multi-level NAND gate circuits

- Design a multi-level circuit of AND- OR gates, the output gate must be OR.
- Two consecutive gates should not be of same type.
- Number the levels starting with the output gate as level 1, replace all gates with NAND gates, leaving all interconnections between gates unchanged, leave the inputs to levels 2,4,6,…..unchanged, invert any literals which appear as inputs to levels 1,3,5……..

Example:



Problem: Find a circuit of NAND gates to realize

F(a,b,c,d) = ∑m(1,5,6,10,13)

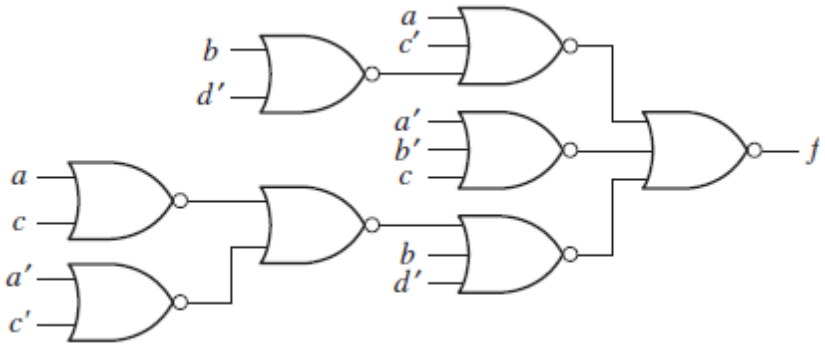
Problem: Realize the following function using three input **Nand- gates**

Z=(AB + C) (D + E + FG) + H

Procedure to design of Multi-level NOR gate circuits

- Design a multi-level circuit of AND- OR gates, the output gate must be AND.
- Two consecutive gates should not be of same type.
- Number the levels starting with the output gate as level 1, replace all gates with NOR gates, leaving all interconnections between gates unchanged, leave the inputs to levels 2,4,6,…..unchanged, invert any literals which appear as inputs to levels 1,3,5……

Problem: Realize f(a,b,c,d) = ∑m(0,3,4,5,8,9,10,14,15) using three-input NOR gates.



Problem: Realize the function given below, using only two-input NAND gates and inverters.

$$f_1 = b'c' + ab' + a'b$$
$$f_2 = b'c' + bc + a'b$$
$$f_3 = a'b'c + ab + bc'$$

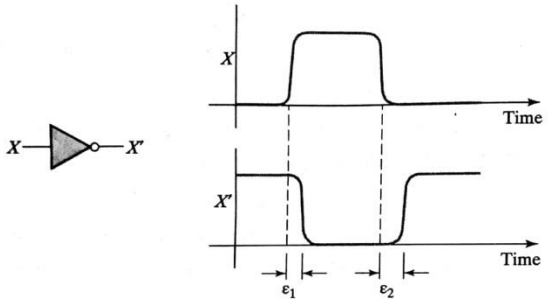
Gate Delays and timing diagrams:

Gate Delay:

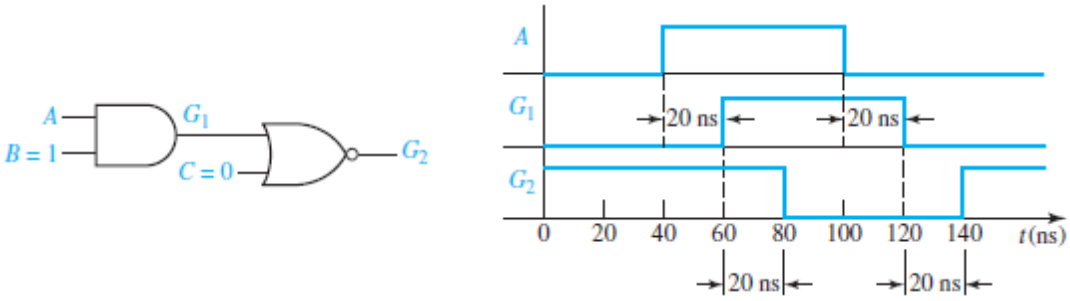
The amount of time taken by gate to react to a change in input is called as gate delay.

Propagation Delay in an Inverter:

Figure below shows input and output waveforms for an inverter, if the change in output is delay by time **Ɛ,** this delay is called as propagation delay.

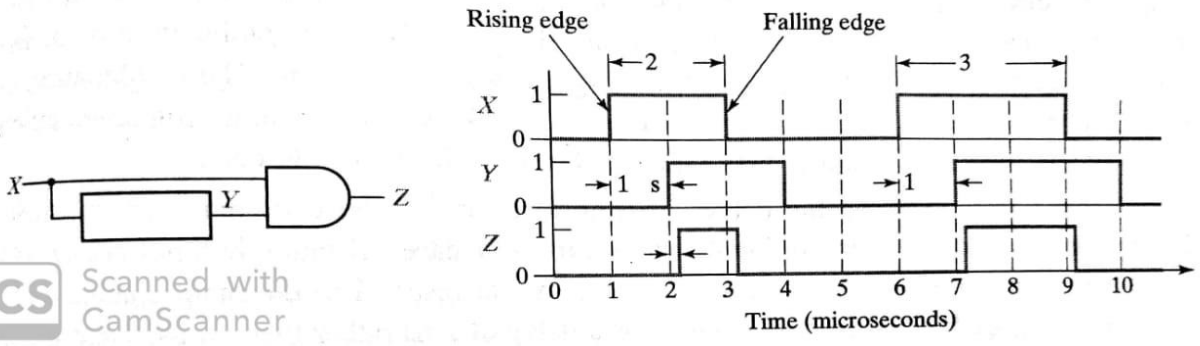


Timing diagram for AND-NOR circuit:



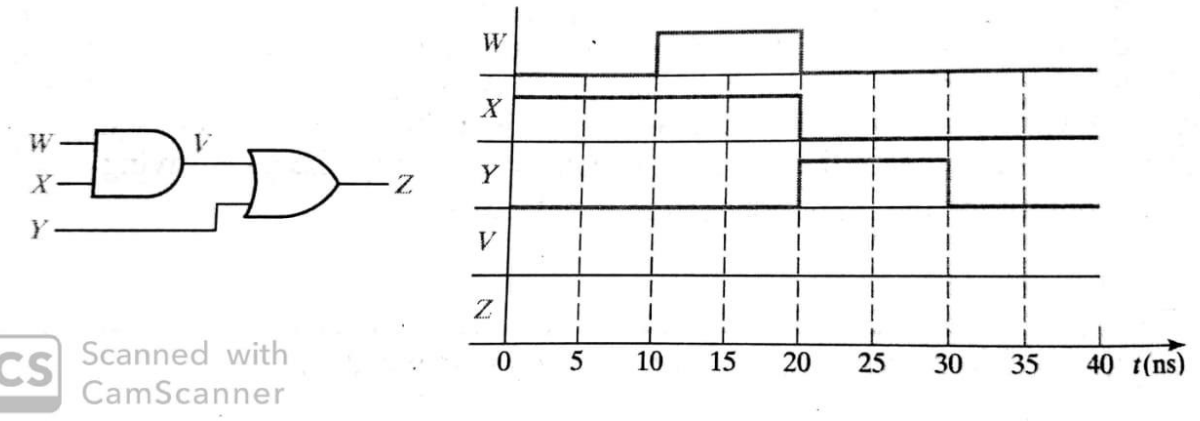
In the above circuit, assume each gate has a propagation delay of 20ns. The timing diagram shows what happens when gate inputs **B** and **C** held at constant values **1 and 0,** and an input A is changed to 1 at **t=40ns** and then changed back to 0 at **t=100ns.** The output of gate G1 changes 20ns after A changes, and the output of gate **G2** changes 20ns after G1 changes.

Timing Diagram for circuit with delay:



The above figure shows the timing diagram for a circuit with an added delay element. The input **X** consists of two pulses, the first of which is 2 microseconds wide and the second is 3 microseconds wide, the delay element has an output **Y** which is the same as the input except that it is delayed by 1 microsecond. The output (**Z**) of the AND gate should be 1 during the time interval in which both **X** and **Y are 1.** if we assume a small propagation delay in the AND gate (**Ɛ**), then **Z** will be as shown in the above figure.

Problem: Complete the timing diagram for the given circuit. Assume that both gates have a propagation delay of 5ns.



<u>Hazards in combinational Logic:</u>

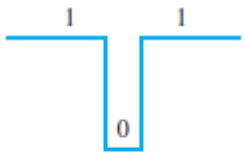
An unwanted transition in output signal of digital circuit is called as **Hazard.**

There are three types of Hazards

1. **Static-1 Hazard**
2. **Static-0 hazard**
3. **Dynamic hazard**

Static-1 hazard:

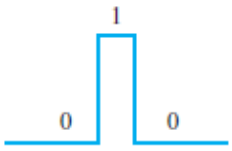
If a circuit output goes to 0 when it should remain a constant 1, we say that circuit has a static-1 hazard.



(a) Static 1-hazard

Static-0 hazard:

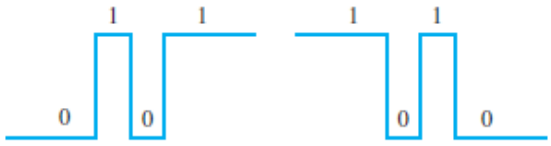
If a circuit output goes to 1 when it should remain a constant 0, we say that circuit has a static-0 hazard.



(b) Static 0-hazard

Dynamic hazard:

When output is supposed to change from **0 to 1 (or 1 to 0)** the output may change three or more times, we say that the circuit has a dynamic hazard.



(c) Dynamic hazards

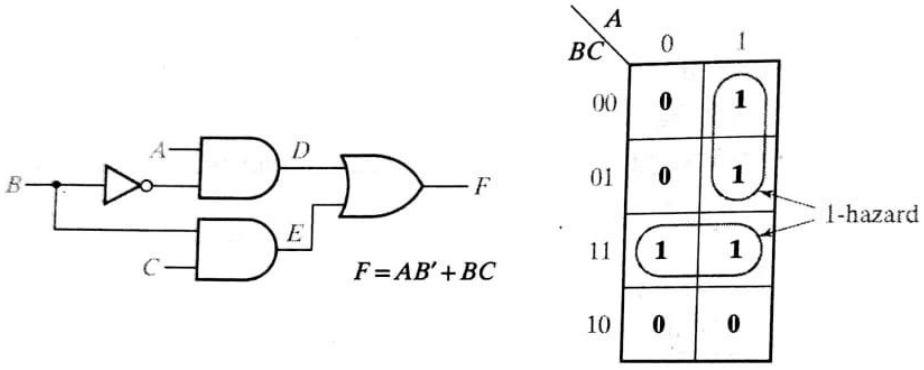
Circuit with Static-1 Hazard:

Figure below shows the logic circuit with static-1 hazard, if **A=C=1,** then **F = B + B'= 1,** so the **F output should** remain a constant 1 when B changes from 1 to 0. As shown in figure(b), if each gate has a

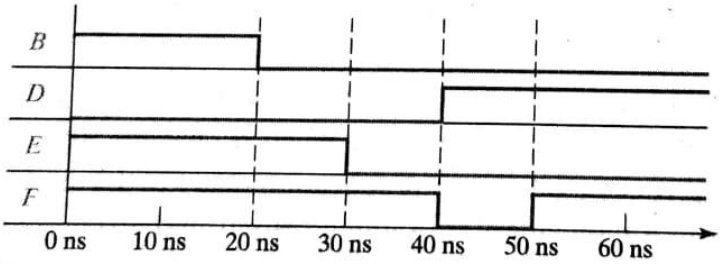
propagation delay of **10ns, E** will go to 0 before **D** goes to 1, resulting in a momentary 0(**glitch) appearing** at the output F.

We can detect hazards in a two-level **AND – OR circuit,** using following procedure.

1. Write down the sum-of products expression for the circuit.
2. Plot each term on the map and loop it.
3. If any two adjacent 1's are not covered by the same loop, a 1-hazard exists for the transition between the two 1's.



(a) Circuit with a static 1-hazard



(b) Timing chart

If we add a loop to the map and then, add the corresponding gate to the circuit, this eliminates the hazard. The term **AC remains** 1 while B is changing, so no glitch can appear in the output.

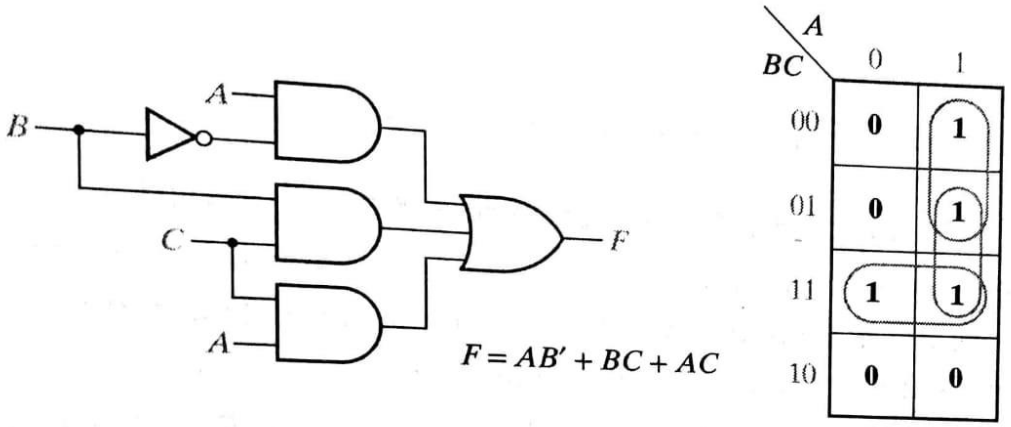
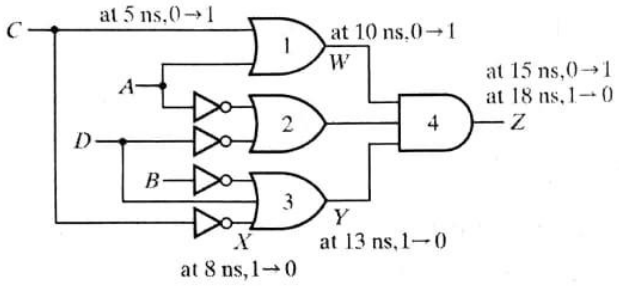
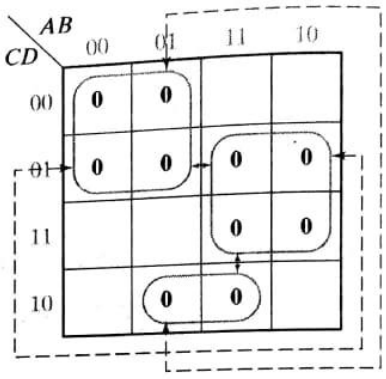


FIGURE 8-9: Circuit with Hazard Removed

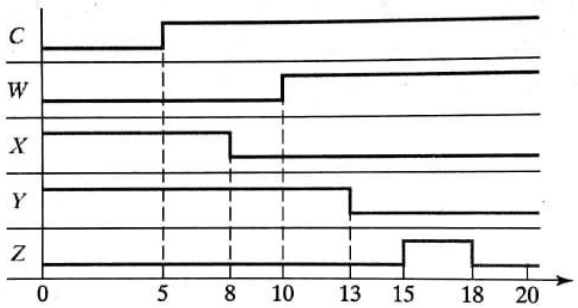
Circuit with static-0 hazard:



(a) Circuit with a static 0-hazard



(b) Karnaugh map for circuit of (a)



(c) Timing diagram illustrating 0-hazard of (a)

The above circuit has **static-0 hazards,** the product of sums representation for the circuit output is

F = (A + C) (A' + D') (B' + C' + D)

The K-map has four pairs of adjacent 0s that are not covered by a common loop as indicated by the arrows, each of these pairs correspond to a 0-hazard. For example when **A =0, B=1 and D=0 and C** changes from 0 to 1, a spike may appear at the output for some combination of gate delays, the timing diagram shows this, assuming gate delays of **3ns for each inverter** and **5ns for each AND gate and OR gate.**

We can eliminate the 0-hazards by looping additional prime Implicant that cover the adjacent 0s that are not already covered by a common loop , this requires three additional loops as shown in figure below, the resulting equation is

F = (A + C) (A' + D')(B' + C' + D) (C + D') (A + B' + D) (A' + B'+ C')

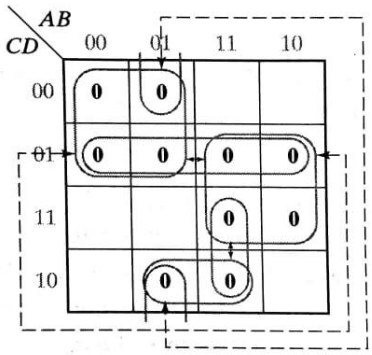
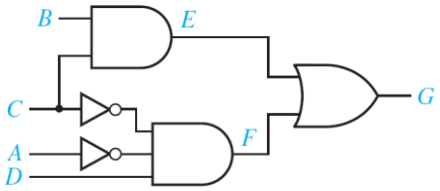


FIGURE 8-11: Karnaugh Map Removing Hazards of Figure 8-10

Problem: For the following circuit

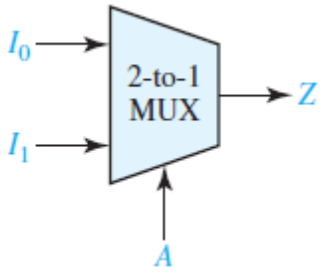
(a) Assume that the inverters have a delay of 1ns and the other gates have a delay of 2ns, initially A=0 and B=C=D=1, and C changes to 0 at time = 2ns, draw a timing diagram and identify the transient that occurs.

(b) Modify the circuit to eliminate the hazard.



Multiplexer (MUX):

- Multiplexer is also called as **Data selector.**
- It has group of **data inputs** and a group of **control inputs.**
- **Control inputs** are used to select one of data inputs and connect it to output terminal.



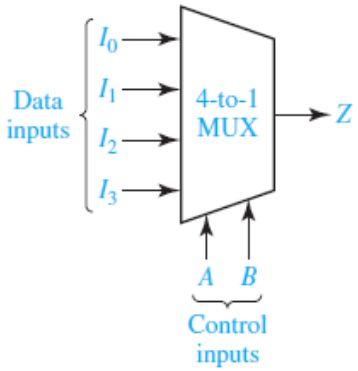
Truth Table

A	Z
0	I_0
1	I_1

2 to 1 MUX Equation:

$$Z = A'I_0 + AI_1$$

4 to 1 Multiplexer:



Truth Table

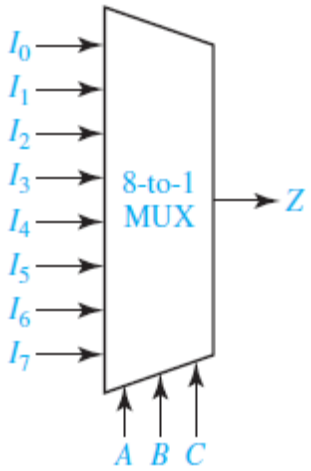
A	B	Z
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

4 to 1 Mux equation:

$$Z = A'B'I_0 + A'BI_1 + AB'I_2 + ABI_3$$

8 to 1 Multiplexer:

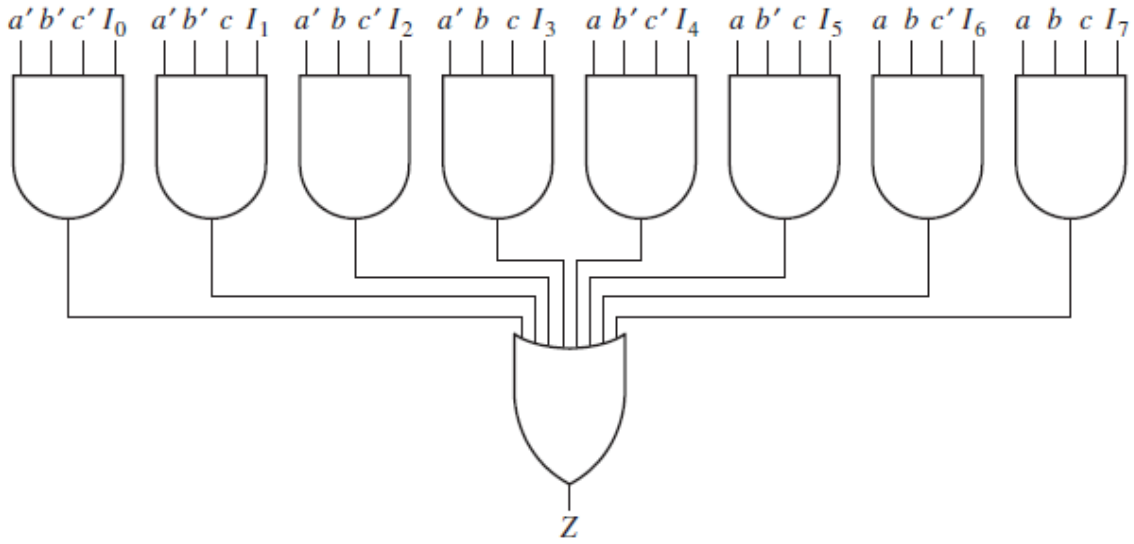
Truth table



A	B	C	Z
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

8 to 1 Mux Equation:

Logic diagram of 8 to 1 Multiplexer:



Multi-level NAND gate Implementation of 8 to 1 Mux:

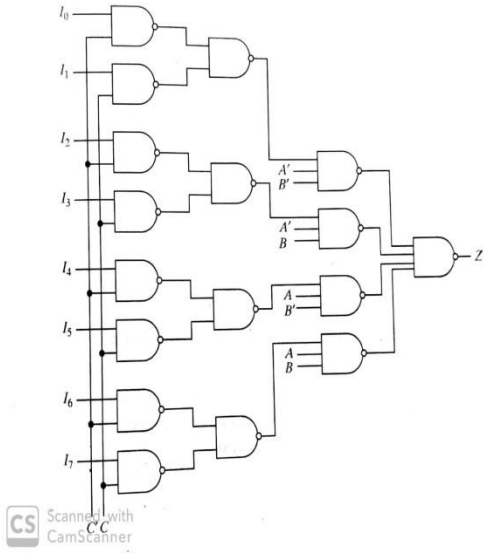
8 to 1 multiplexer equation

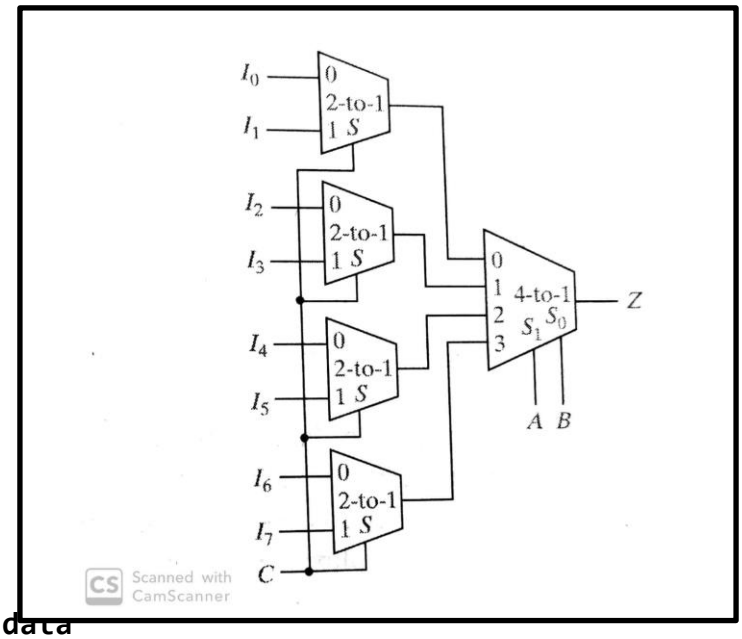
$Z=A'B'C'I_0 + A'B'CI_1 + A'BC'I_2 + A'BCI_3 + AB'C'I_4 + AB'CI_5 + ABC'I_6 + ABCI_7$

Implementation with more than two levels of gates can be obtained by factoring the equation for Z.

$Z=A'B'(C'I_0+CI_1) +A'B(C'I_2+CI_3) +AB'(C'I_4+CI_5) + AB(C'I_6 + CI_7)$

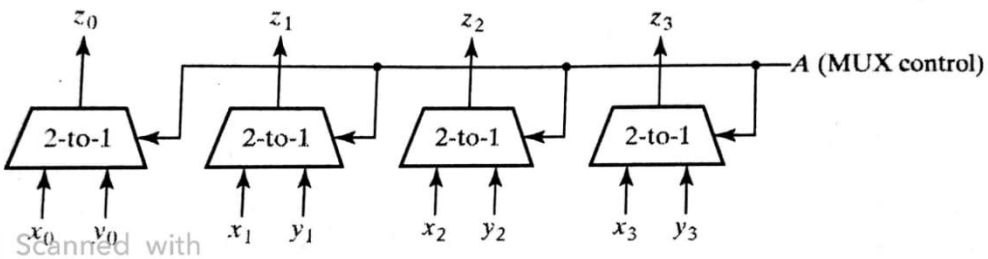
The above equation can be implemented using NAND gates as shown below.



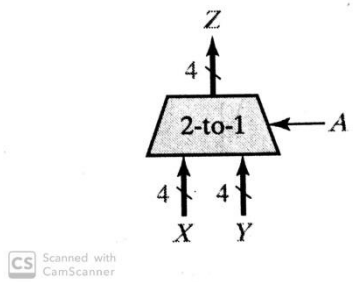


Quad Multiplexer used to select data

The quad multiplexer is used to select one of two 4-bit data words, if control input A=0, the values of **x_0, x_1, x_2, x_3** will appear at **Z_0, Z_1, Z_2 and Z_3** outputs, if **A=1** the values of **Y_0, Y_1, Y_2 and Y_3** will appear at the outputs.

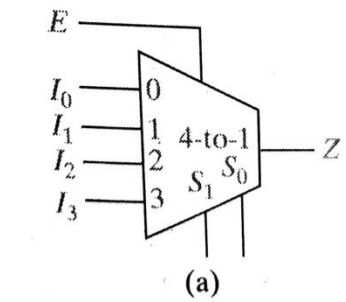


Quad multiplexer with bus inputs and output



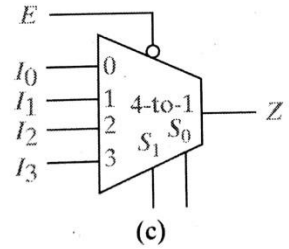
Multiplexer with Active high Enable input:

Multiplexer has an additional input called **Enable.** If E=0 irrespective of data inputs and control inputs output Z=0, if E=1 then multiplexer output depends on control inputs and data inputs.



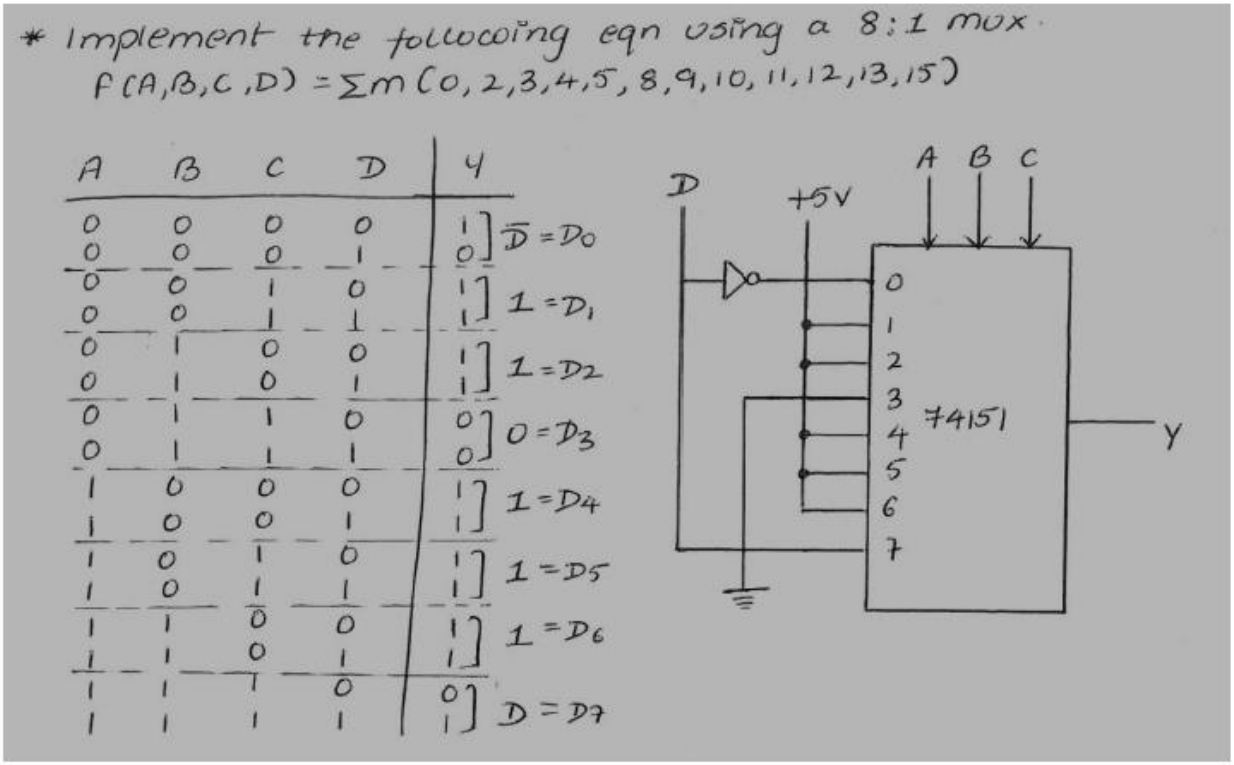
E	S1	S0	Z
0	X	X	0
1	0	0	I_0
1	0	1	I_1
1	1	0	I_2
1	1	1	I_3

Multiplexer with Active LOW Enable input:



E	A	B	Z
1	X	X	0
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3

Problem:Implement the function Z = A'C'D'+B'C'D'+A'C'D+AB'CD'+A'BCD' using 4 to 1 multiplexer.



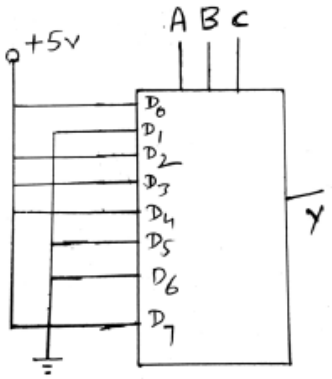
Problem: Realize a) **$Y=A'B + B'C'+ABC$ using 8 to 1 multilexer, b) can it be realized with 4 to 1 multiplexer?**

Sol) convert the given equation into 8 to 1 multiplexer logic equation

$$Y= A'B + B'C' + ABC$$
$$Y = A'B(C+C') +B'C' (A+A') + ABC$$
$$Y= A'BC+A'BC' + B'C'A+B'C'A' + ABC$$
$$Y= A'B'C' .1 + A'B'C .0 + A'BC' .1 + A'BC.1+AB'C' .1 + AB'C .0 + ABC' .0 +ABC.1$$

From the above equation D0 = D2= D3=D4=D7=1 D1= D5=D6=0

Circuit diagram:

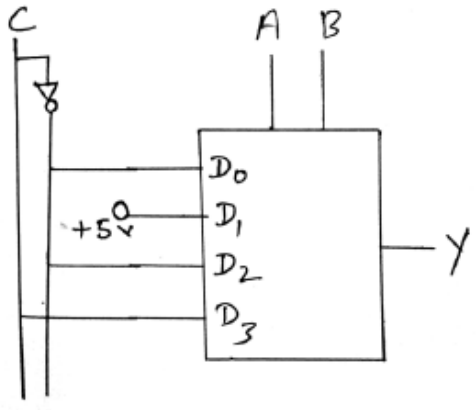


b) yes, the equation **$Y= A'B + B'C' + ABC$** can be realized with 4 to 1 multiplexer

the above equation can be written as

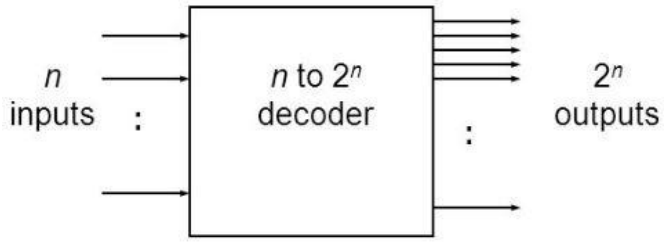
$$Y= A'B + B'C'(A+A') + ABC$$
$$Y= A'B' C' + A'B .1 + AB' C' + AB C$$

From the above equation **D0 = D2=C' D1=1 D3=C**

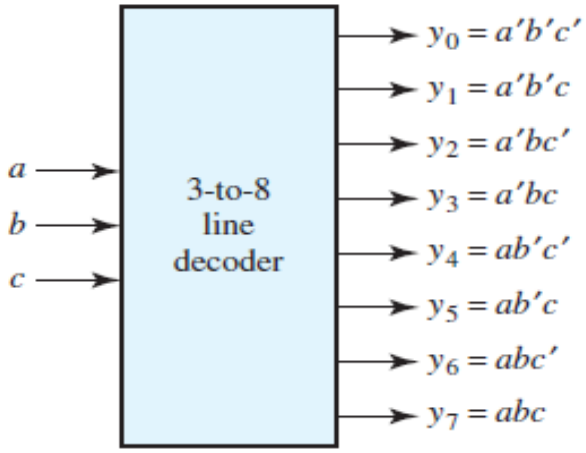


Decoder:

Decoder is a logic circuit with n-inputs and 2^n outputs.



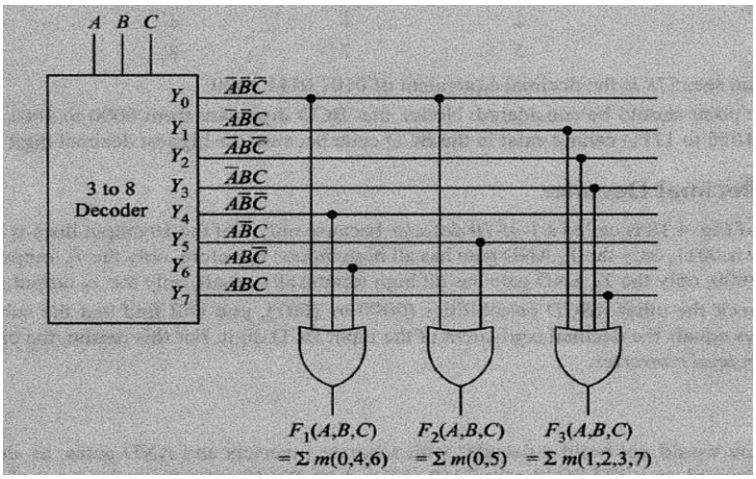
3 to 8 Decoder:



a	b	c	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
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

Problem: implement the following functions using 3 to 8 decoder and OR gates

$F_1(A,B,C) = \sum m(0,4,6)$
$F_2(A,B,C) = \sum m(0,5)$
$F_3(A,B,C) = \sum m(1,2,3,7)$

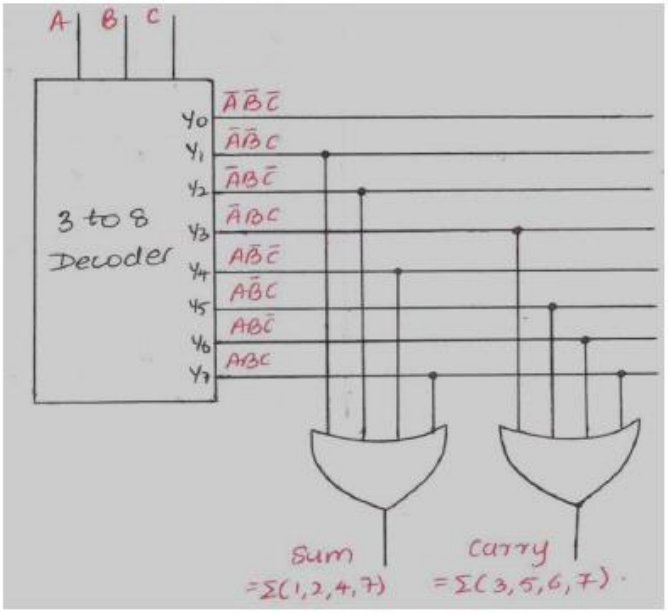


Problem: implement full adder using 3 to 8 decoder

Full adder truth table:

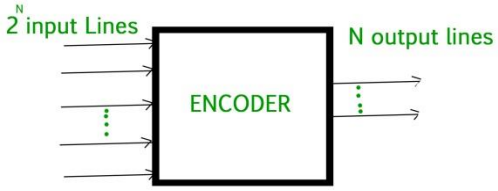
A	B	C	Sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Full adder using 3 to 8 decoder



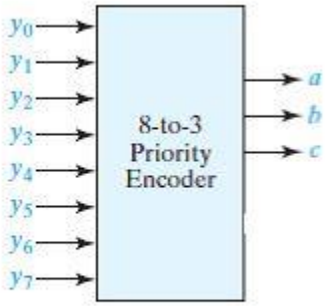
Encoder:

- An encoder converts active input signal into coded output signal.
- It has 2^n inputs and n outputs.
- One of 'n' outputs is active at any given time.



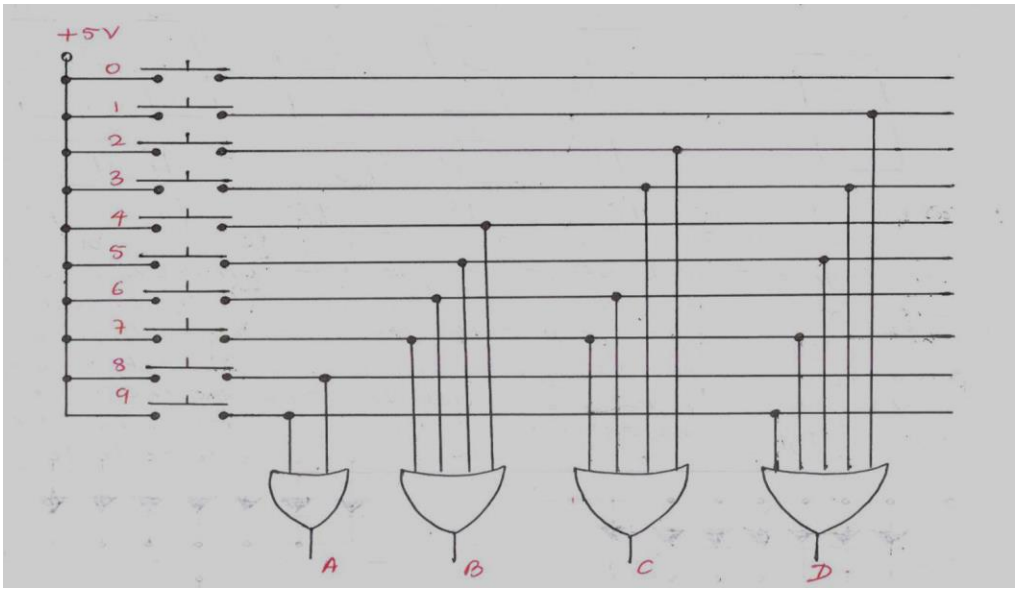
8 to 3 Priority Encoder:

An 8-to-3 Priority Encoder



y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	a	b	c
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
X	1	0	0	0	0	0	0	0	0	1
X	X	1	0	0	0	0	0	0	1	0
X	X	X	1	0	0	0	0	0	1	1
X	X	X	X	1	0	0	0	1	0	0
X	X	X	X	X	1	0	0	1	0	1
X	X	X	X	X	X	1	0	1	1	0
X	X	X	X	X	X	X	1	1	1	1

Problem: Design Decimal to BCD Encoder



Programmable Logic Devices:

Programmable logic devices are used to implement logic functions, there are two types of programmable logic devices

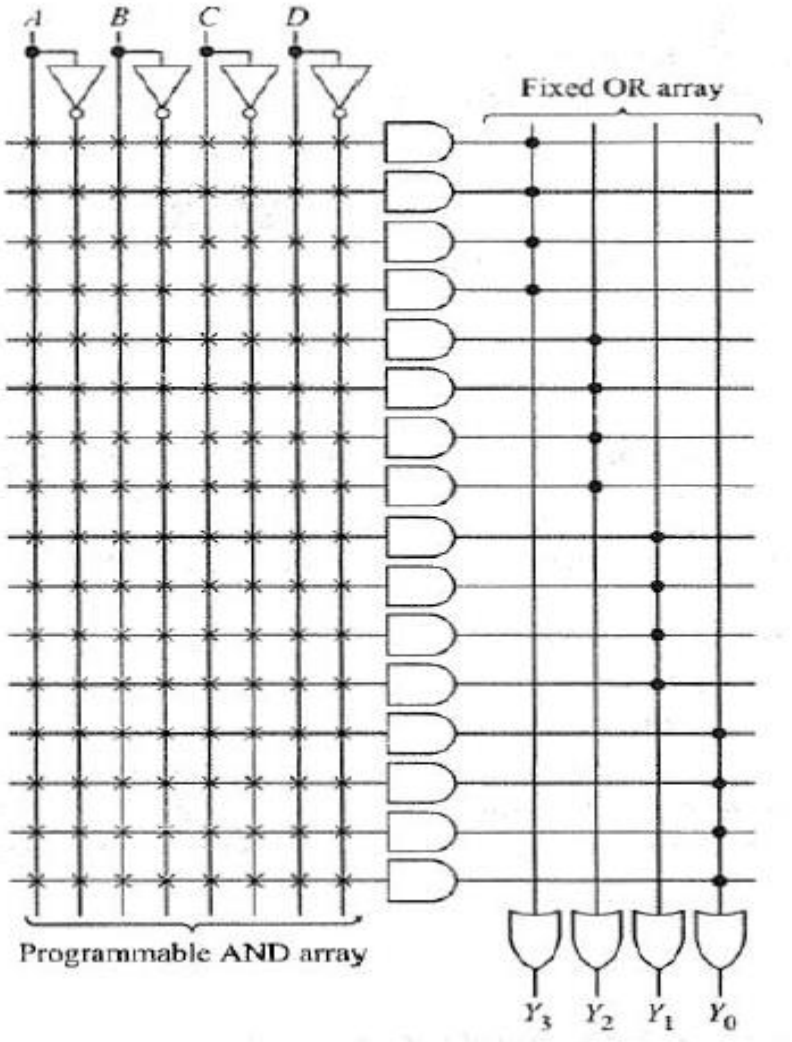
1. PAL (Programmable Array Logic)
2. PLA (Programmable Logic Array)

PAL (Programmable Array Logic)

- In PAL, **AND** array is programmable and the **OR array is fixed.**

Structure of 4-input and 4-output PAL:

In the figure, array of 16 **AND** gates are used, and array of four **OR** gates are used, symbol '×' is fusible link and **dot(·)** is fixed connection the output of first four AND gates are connected as inputs to OR gate Y_3, and outputs of next four AND gates are connected as inputs to OR gate Y2 and so on



Problem: Implement full adder using PAL

Problem: Implement the following function using PAL

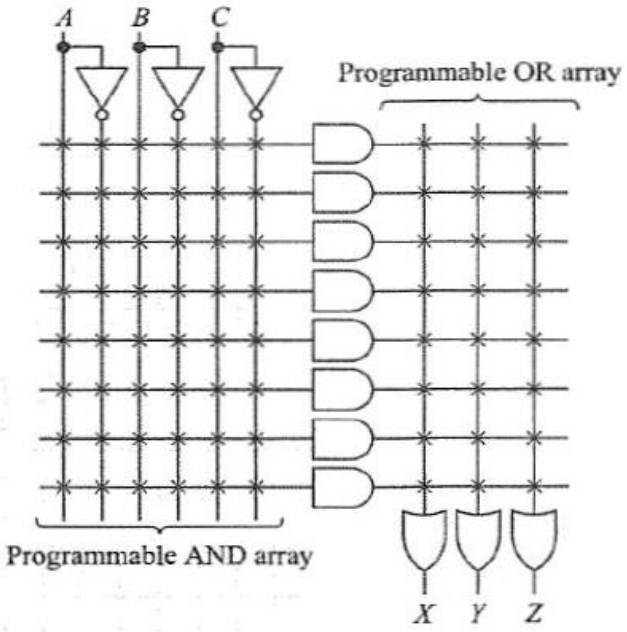
F(A,B,C,D) = ∑m(0,2,6,7,8,9,12,13)

Programmable Logic Array:

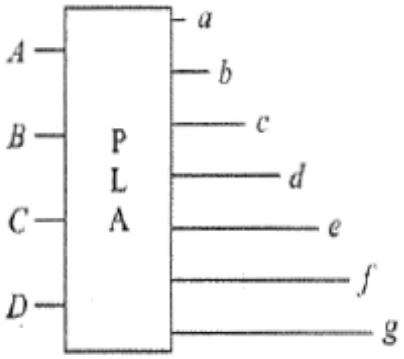
PLA has programmable AND array and programmable OR array, PLA is more complicated to use since the number of fusible links are more.

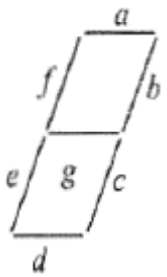
Structure of 3-input 3-output PLA:

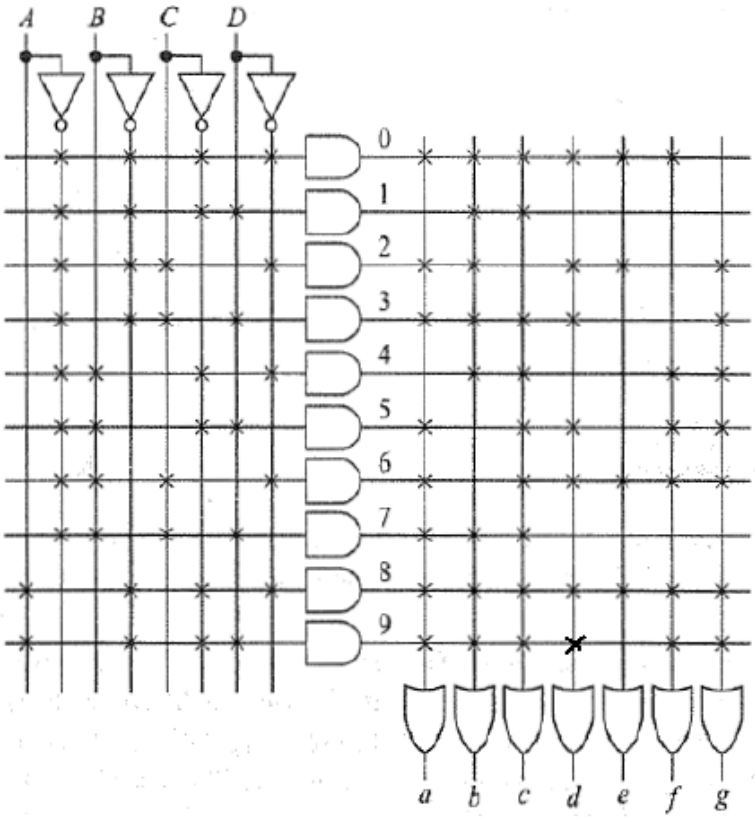
As shown in figure, it has three inputs A,B,C And three output X,Y,Z, Array of eight AND Gates are used and array of 3 OR gates are used



Problem: Implement a 7-segmnet display using PLA







BCD – Seven Segment Display converter table:

Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	0	1	0	1	1	0	0	0	0	1
2	0	0	1	0	1	1	0	1	1	0	1	2
3	0	0	1	1	1	1	1	1	0	0	1	3
4	0	1	0	0	0	1	1	0	0	1	1	4
5	0	1	0	1	1	0	1	1	0	1	1	5
6	0	1	1	0	1	0	1	1	1	1	1	6
7	0	1	1	1	1	1	1	0	0	0	0	7
8	1	0	0	0	1	1	1	1	1	1	1	8
9	1	0	0	1	1	1	1	1	0	1	1	9

Important Questions

1. Give differences between combinational and sequential circuits.

2. Find a circuit of NAND gates to realize F(A,B,C,D)=∑m(1,5,6,10,13)

3. Realize the following function using three input NAND gates

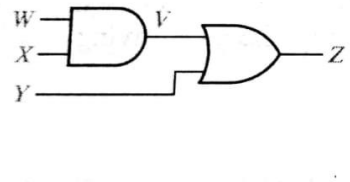
   Z = (AB + C)(D + E + FG) + H

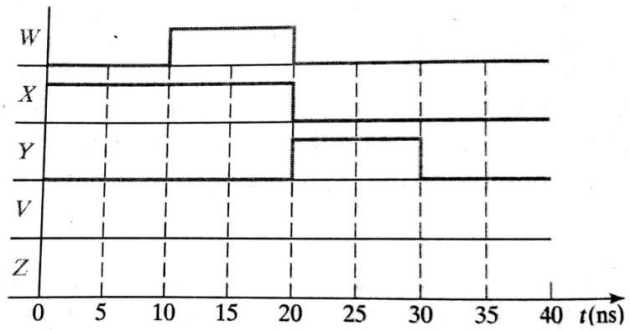
4. Realize f(a,b,c,d) = ∑m(0,3,4,5,8,9,10,14,15) using three-input NOR gates.

5. Realize the function given below, using only two-input NAND gates and inverters.

$$f_1 = b'c' + ab' + a'b$$
$$f_2 = b'c' + bc + a'b$$
$$f_3 = a'b'c + ab + bc'$$

6. Complete the timing diagram for the given circuit. Assume that both gates have a propagation delay of 5ns.



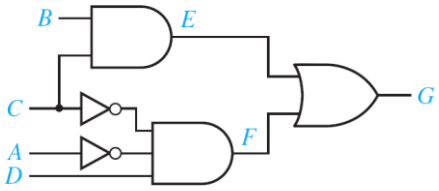


7. Define hazard and explain types of hazards.

8. Problem: For the following circuit

(a) Assume that the inverters have a delay of 1ns and the other gates have a delay of 2ns, initially A=0 and B=C=D=1, and C changes to 0 at time = 2ns, draw a timing diagram and identify the transient that occurs.

(b) Modify the circuit to eliminate the hazard.



9. Implement 8 to 1 MUX using multi-level NAND gates.

10. Draw and explain QUAD Multiplexer with neat diagram.

11. Implement the function Z = A'C'D'+B'C'D'+A'C'D+AB'CD'+A'BCD' using 4 to 1 multiplexer.

12. Realize **Y=A'B +B'C'+ABC** using 8 to 1 multiplexer b) can it be realized with 4 to 1 Multiplexer.

13. Problem: implement the following functions using 3 to 8 decoder and OR gates

$F_1(A,B,C) = \sum m(0,4,6)$
$F_2(A,B,C) = \sum m(0,5)$
$F_3(A,B,C)=\sum m(1,2,3,7)$

14. Implement full adder using 3 to 8 decoder

15. Draw and explain 8 to 3 Priority Encoder.

16. Implement Full Adder using 3 to 8 Decoder.

17. Design Decimal to BCD Encoder.

18. Implement full adder using PAL

19. **Implement the following function using PAL**

$F(A,B,C,D) = \sum m(0,2,6,7,8,9,12,13)$

20. Draw a structure of three input three outputs PLA

21. Implement 7 segment display using PLA