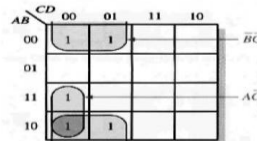
CHAPTER FIVE



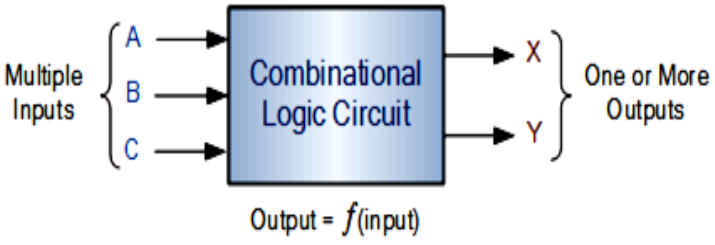
COMBINATIONAL LOGIC CIRCUITS, ANALYSIS AND DESIGN PROCEDURE.

The following points are the Objectives of the present chapter:

- Analyze basic combinational logic circuits, such as AND-OR, AND-OR-Invert, exclusive-OR, and exclusive-NOR.
- Use AND-OR and AND-OR-Invert circuits to implement sum-of-products (SOP) and product-of sums (POS) expressions.
- Write the Boolean output expression for any combinational logic circuit.
- Develop a truth table from the output expression for a combinational logic circuit.
- Use the Karnaugh map to expand an output expression containing terms with missing variables into a full SOP form.
- Design a combinational logic circuit for a given Boolean output expression.
- Design a combinational logic circuit for a given truth table.
- Simplify a combinational logic circuit to its minimum form.
- Use NAND gates to implement any combinational logic function.

Combinational Logic Circuits are memoryless digital logic circuits whose output at any instant in time depends only on the combination of its inputs.

Or the combinational logic circuits or time-independent logic circuits in digital circuit theory can be defined as a type of digital logic circuit implemented using Boolean circuits, where the output of logic circuit is a pure function of the present inputs only.

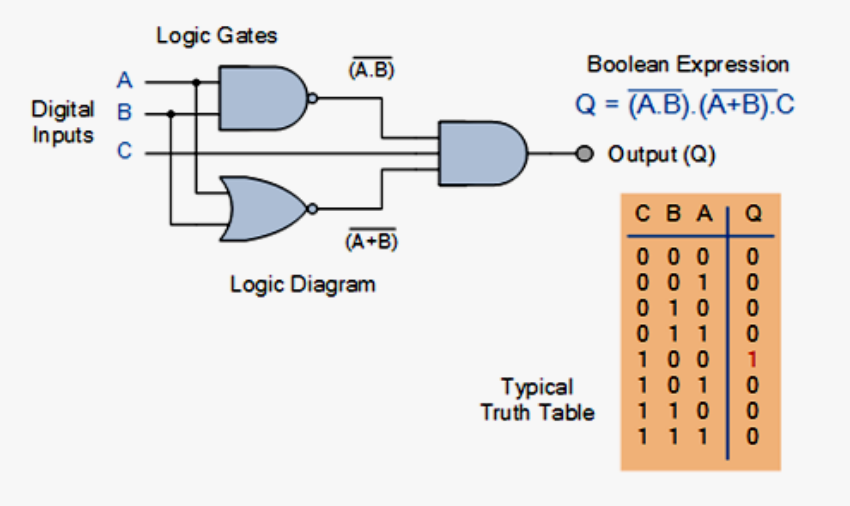


Combinational Logic Circuits are made up from basic logic NAND, NOR or NOT gates that are "combined" or connected together to produce more complicated switching circuits. These logic gates are the building blocks of combinational logic circuits.

Combinational logic circuits can be very simple or very complicated and any combinational circuit can be implemented with only NAND and NOR gates as these are classed as "universal" gates.

The three main ways of specifying the function of a combinational logic circuit are:

1. Boolean Algebra – This forms the algebraic expression showing the operation of the logic circuit for each input variable either True or False that results in a logic "1" output.
2. Truth Table – A truth table defines the function of a logic gate by providing a concise list that shows all the output states in tabular form for each possible combination of input variable that the gate could encounter.
3. Logic Diagram – This is a graphical representation of a logic circuit that shows the wiring and connections of each individual logic gate, represented by a specific graphical symbol, that implements the logic circuit.



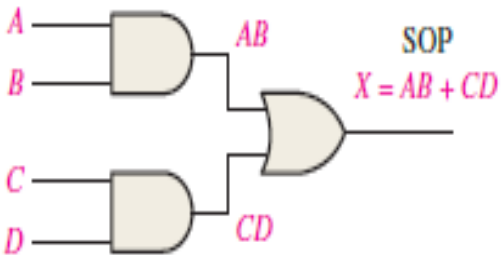
5.1. Basic Combinational Logic Circuits.

- In Chapter 4, we learned that SOP expressions are implemented with an AND gate for each product term and one OR gate for summing all of the product terms.
- This SOP implementation is called AND-OR logic and is the basic form for realizing stranded Boolean functions.

- In this section, the AND-OR and the AND-OR-inverter are examined; the exclusive-OR and exclusive-NOR gates, which are actually a form of AND-OR logic, are also covered.

5.1.1. AND-OR Logic.

- The Figure shows an AND-OR circuit consisting of two 2-input AND gates and one 2-input OR gate.



- The Boolean expressions for the AND gate outputs and the resulting SOP expression for the output X are shown on the diagram.
- In general, all AND-OR circuit can have any number of AND gates each with any number of inputs.
- The truth table for a 4-input AND-OR logic circuit is shown in the Table.
- The operation of the AND-OR circuit shown in figure is stated as follows: For a 4-input AND-OR logic circuit, the output X is HIGH (1) if both input A and input B are HIGH (1) or both input C and input D are HIGH (1).

Truth Table

INPUT						OUTPUT
A	B	C	D	AB	CD	X
0	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	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	1	1
1	1	0	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	0	1
1	1	1	0	1	0	1

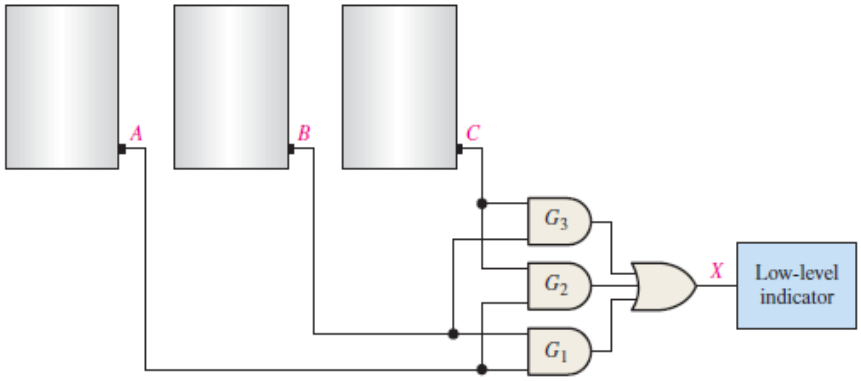
Example (1):

In a certain chemical-processing plant, a liquid chemical is used in a manufacturing process. The chemical is stored in three different tanks. A level sensor in each tank produces a HIGH voltage when the level of chemical in the tank drops below a specified point.

Design a circuit that monitors the chemical level in each tank and indicates when the level in any two of the tanks drops below the specified point.

Solution

The AND-OR circuit in Figure below has inputs from the sensors on tanks *A*, *B*, and *C* as shown. The AND gate *G*1 checks the levels in tanks *A* and *B*, gate *G*2 checks tanks *A* and *C*, and gate *G*3 checks tanks *B* and *C*. When the chemical level in any two of the tanks gets too low, one of the AND gates will have HIGHs on both of its inputs, causing its output to be HIGH; and so the final output *X* from the OR gate is HIGH. This HIGH input is then used to activate an indicator such as a lamp or audible alarm, as shown in the figure.

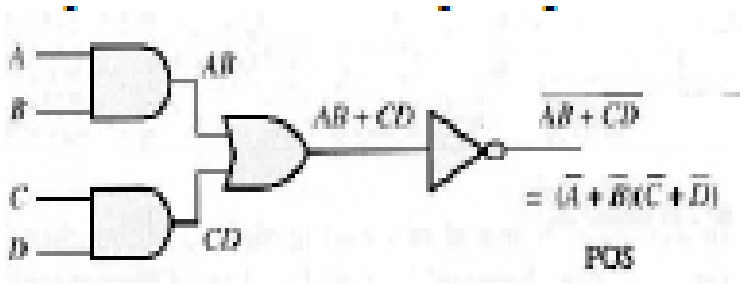


5.1.2. AND-OR-Invert Logic.

- When the output of an AND-OR circuit is complemented (inverted), it results in an AND-OR-Invert circuit.
- Recall that AND-OR logic directly implements SOP expressions.
- POS expressions can be implemented with AND-OR-Invert logic.
- This is illustrated as follows, starting with a POS expression and developing the corresponding AND-OR-Invert expression.

$$X = (\overline{A} + \overline{B})(\overline{C} + \overline{D})$$
$$X = (\overline{AB})(\overline{CD})$$
$$X = \overline{(\overline{AB})(\overline{CD})}$$
$$X = \overline{(\overline{\overline{AB}}) + (\overline{\overline{CD}})}$$
$$X = \overline{(AB) + (CD)}$$

- The logic diagram in the Figure shows an AND-OR-Invert circuit and the development of the POS output expression.



- In general, an AND-OR-Invert circuit can have any number of AND gates each with any number of inputs.
- The operation of the AND-OR-Invert circuit is stated as follows: For a 4-input AND-OR-Invert logic circuit, the output X is LOW

(0) if both input A and input B are HIGH (1) or both input C and input D are HIGH (1).

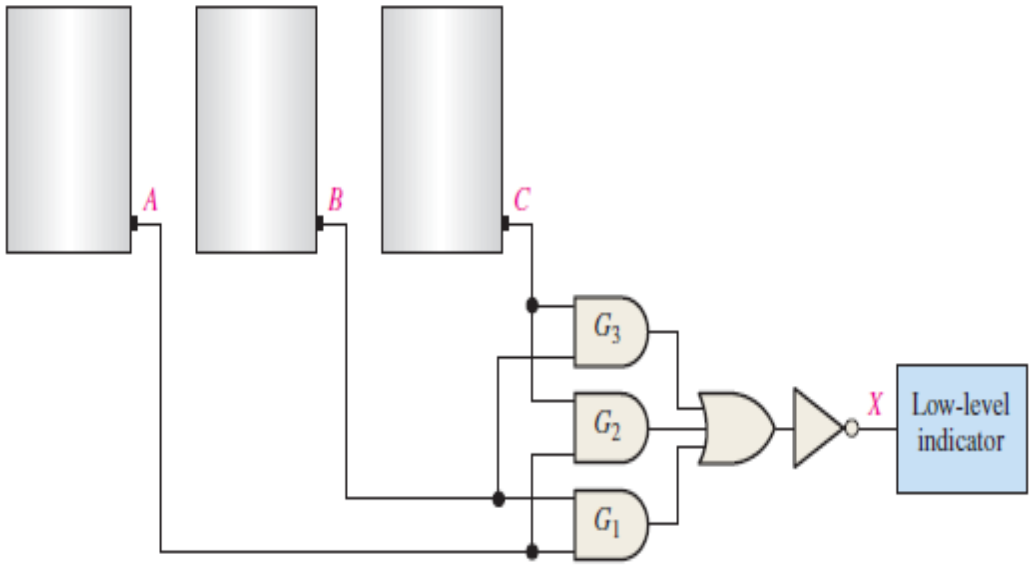
- **A truth table of AND-OR-Invert can be developed from the AND- OR truth table by simply changing all 1s to 0s and all 0s to 1s in the output column.**

Example (2):

The sensors in the chemical tanks of Example (1) are being replaced by a new model that produces a LOW voltage instead of a HIGH voltage when the level of the chemical in the tank drops below a critical point. Modify the circuit in the above figure to operate with the different input levels and still produce a HIGH output to activate the indicator when the level in any two of the tanks drops below the critical point. Show the logic diagram.

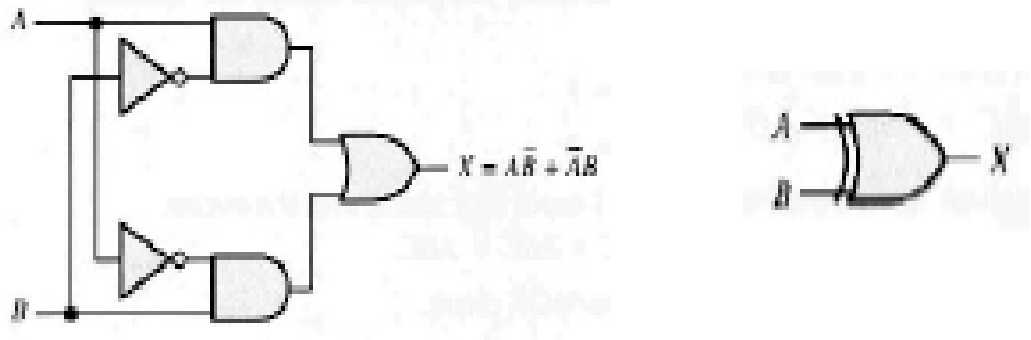
Solution

The AND-OR-Invert circuit has inputs from the sensors on tanks *A*, *B*, and *C* as shown below. The AND gate *G*1 checks the levels in tanks *A* and *B*, gate *G*2 checks tanks *A* and *C*, and gate *G*3 checks tanks *B* and *C*. When the chemical level in any two of the tanks gets too low, each AND gate will have a LOW on at least one input, causing its output to be LOW and, thus, the final output *X* from the inverter is HIGH. This HIGH output is then used to activate an indicator.



5.1.3. Exclusive-OR logic.

- The exclusive-OR gate was studied previously. Although, because of its importance, this circuit is considered a type of logic gate with its own unique symbol it is actually a combination of two AND gates, one OR gate, and two inverters, as shown in the Figure.



- The output expression for the circuit in the Figure is:

$$X = \bar{A}\,B + A\,\bar{B}$$

- Evaluation of this expression results in the truth table in the Table.

A	B	$A\bar{B}$	$\bar{A}B$	X
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

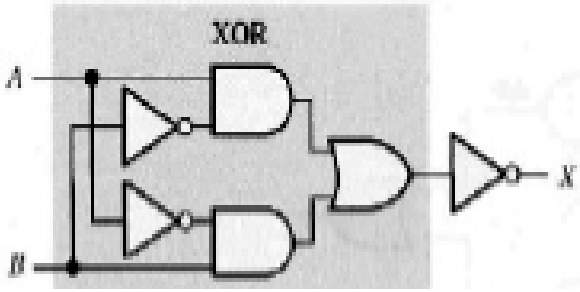
- A special exclusive-OR operator $\oplus$ is often used, so the expression $X = \bar{A}\,B + A\,\bar{B}$ can be stated as "X is equal to A exclusive-OR B" and can be written as $X = A \oplus B$.
- Notice that the output is HIGH only when the two inputs are at opposite levels.

5.1.4. Exclusive-NOR Logic.

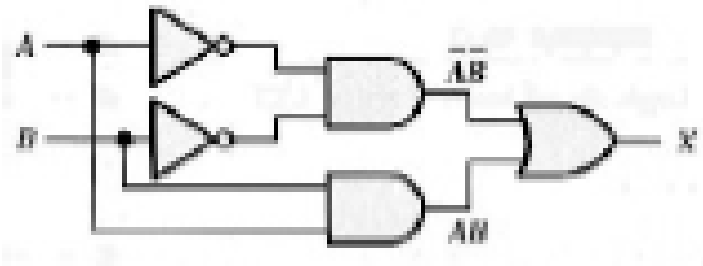
- The complement of the exclusive-OR function is the exclusive-NOR, which is derived as follows:

$$X = \overline{\overline{A}\overline{B} + \overline{A}B}$$
$$X = (A\overline{B})(\overline{A}B)$$
$$X = (\overline{A} + B)(A + \overline{B})$$
$$X = A\overline{A} + \overline{A}\overline{B} + AB + B\overline{B}$$
$$X = AB + \overline{A}\overline{B}$$

- Notice that the output X is HIGH only when the two inputs, A and B, are at the same level.
- The exclusive-NOR can be implemented by simply inverting the output of an exclusive OR, as shown in the Figure.

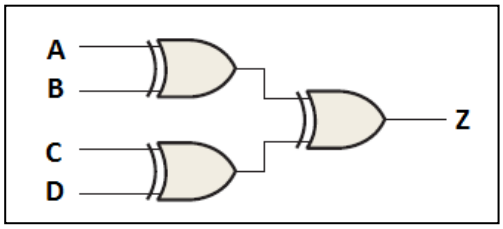


- Or by directly implementing the expression $AB + \overline{A}\,\overline{B}$ as shown in the Figure.



Example (3):
Design a circuit that generate an output=1 if the number of ones on a four input data is odd.

Solution:
The output Z of circuit shown below equals to 1 when the number of ones on the A, B, C, and D input is odd otherwise Z=0.



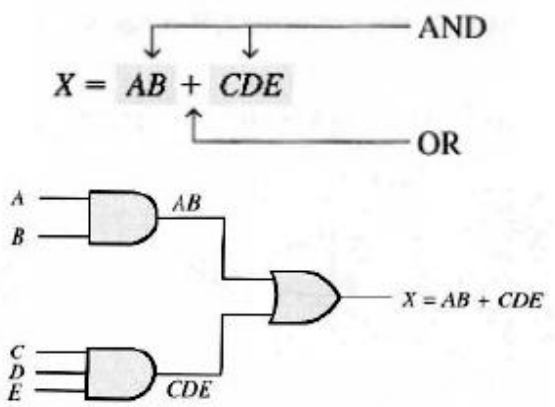
A	B	C	D	Z
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

5.2. Implementing Combinational Logic.

5.2.1.From a Boolean Expression to a Logic Circuit.

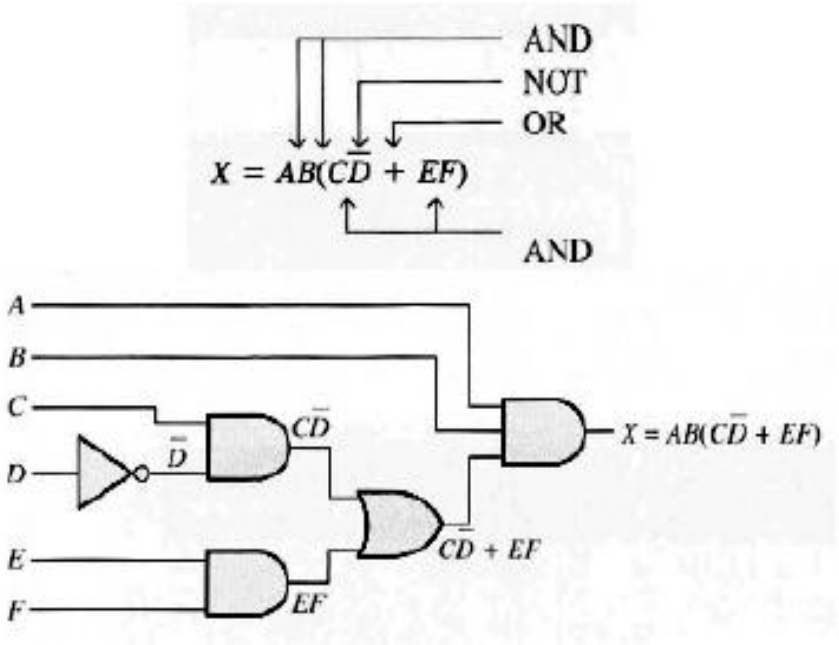
let's examine the following Boolean expression:

$$X = AB + CDE$$



Let's implement the following expression:

$$X = AB (C\overline{D} + EF)$$



The logic gates required to implement $X = AB(C\overline{D} + EF)$ are as follows:

1. One inverter to form $\overline{D}$

2. Two 2-input AND gates to form CD and EF

3. One 2-input OR gate to form $C\overline{D} + EF$

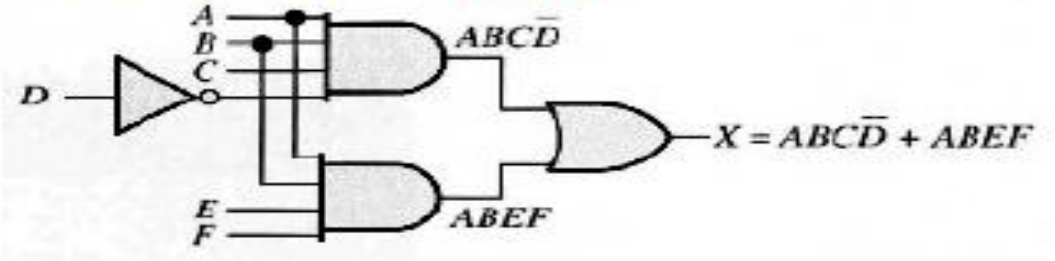
4. One 3-input AND gate to form X

Another Solution: SOP

The expression is converted to SOP as follows, and the resulting expression and circuit are shown below,

$$X = AB(C\overline{D} + EF)$$

$$X = ABC\overline{D} + ABEF$$



The logic gates required to implement X are as follows:

1. One inverter to form $\overline{D}$

2. Two 2-input AND gates to form $ABC\overline{D}$ and $ABEF$

3. One 2-input OR gate to form X

5.2.2.From a Truth Table to a logic Circuit.

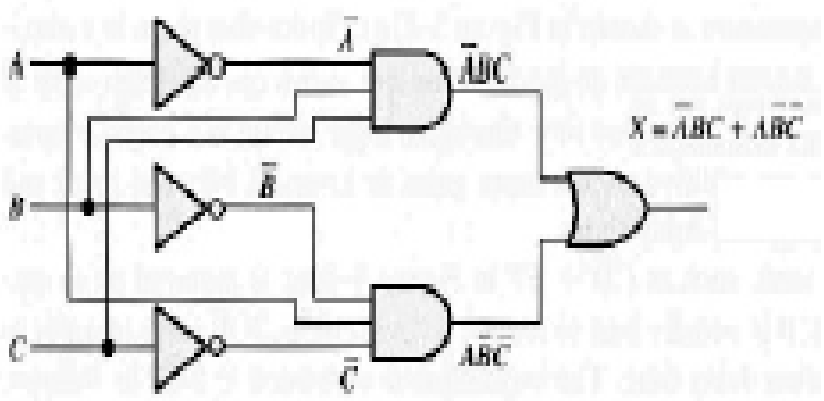
- If you begin with a truth table instead of an expression, you can write the SOP expression from the truth table and then implement the logic circuit.
- The Table shown specifies a logic function.

Truth Table

INPUT			OUTPUT
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

- The Boolean SOP expression obtained from the truth table by ORing the product terms for which X = 1 is:

$$X = \bar{A}BC + A\overline{B}\,\bar{C}$$



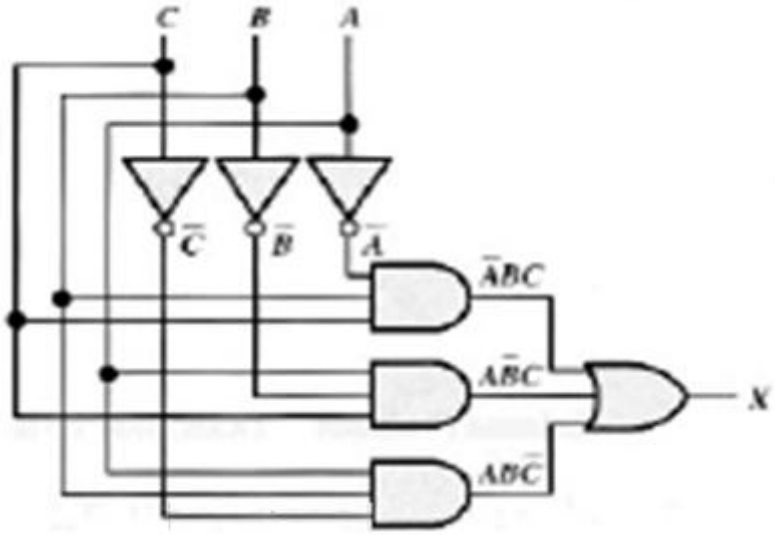
Example (4):

Design a logic circuit to implement the operation specified in the truth table shown.

INPUT			OUTPUT
A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	1
1	0	0	
1	0	1	1
1	1	0	1
1	1	1	

Solution: The logic expression is: $X = \bar{A}\,B\,C + A\,\bar{B}\,C + A\,B\,\bar{C}$

- The logic gates required are three inverters, three 3-input AND gates and one 3-input OR gate.
- The logic circuit is shown in the Figure.



Example (5): Develop a logic circuit with four input variables that will only produce a 1 output when exactly three input variables are 1s.

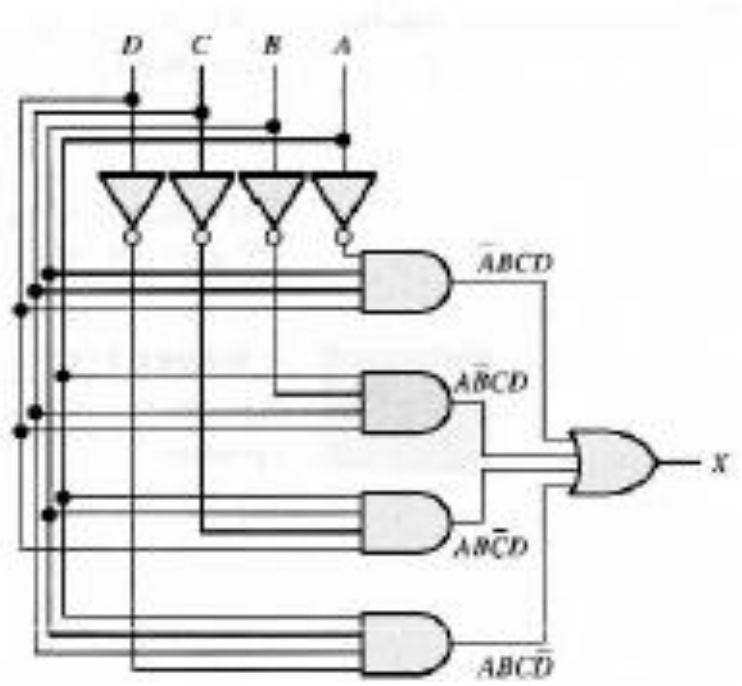
Solution: Out of sixteen possible combinations of four variables, the combinations in which there are exactly three 1s, are listed in the Table.

A	B	C	D	X
0	1	1	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1

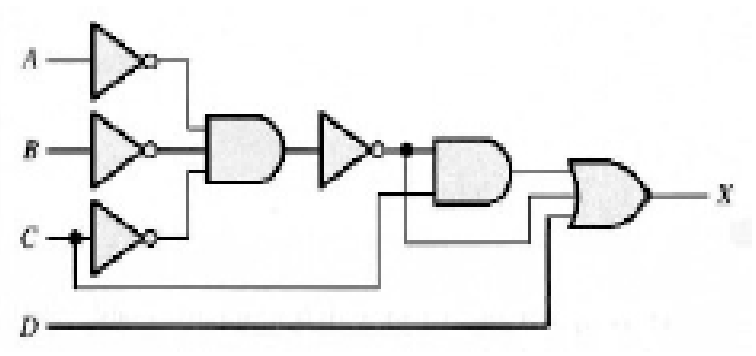
- The product terms are ORed to get the following expression:

$$X = \bar{A}\,B\,C\,D + A\,\bar{B}\,C\,D + A\,B\,\bar{C}\,D + A\,B\,C\,\bar{D}$$

The logic circuit is shown in figure below.



Example (6): Reduce the combinational logic circuit shown in the figure below to a minimum form.



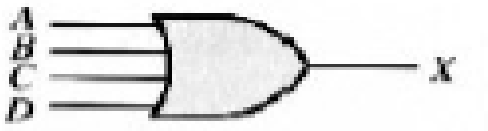
Solution: The expression for the output of the circuit is:

$$X = (\overline{\overline{A}\,\overline{B}\,\overline{C}})C + \overline{\overline{A}\,\overline{B}\,\overline{C}} + D$$

- Applying DeMorgan's theorem and Boolean algebra;

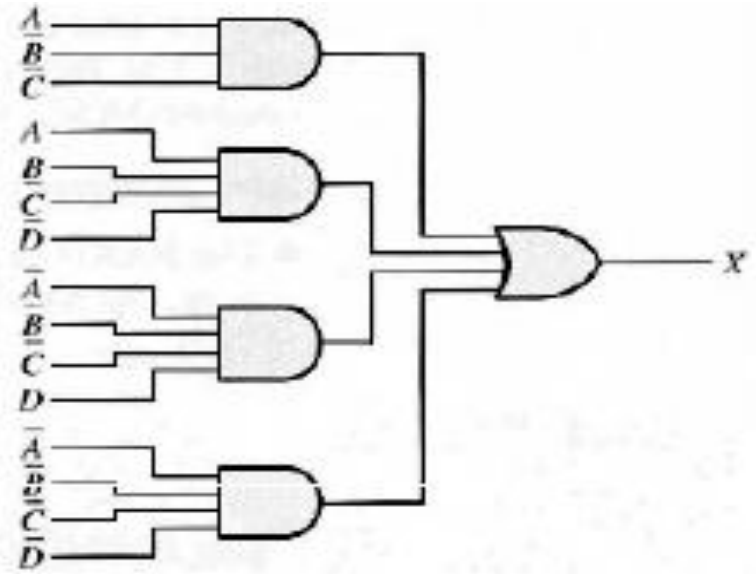
$$\begin{aligned}
&X = (\overline{\overline{A} + \overline{B} + \overline{C}}) \times C + (\overline{\overline{A} + \overline{B} + \overline{C}}) + D \\
&X = (A + B + C) \times C + (A + B + C) + D \\
&X = AC + BC + CC + A + B + C + D \\
&X = AC + BC + C + A + B + C + D \\
&X = AC + BC + C + A + B + D \\
&X = C(A + B + 1) + A + B + D \\
&X = A + B + D + C
\end{aligned}$$

- The simplified circuit is a 4-input OR gate as shown in the Figure.



5.2.3.From Logic diagram and Karnaugh Map simplification.

Example (7): Minimize the combinational logic circuit in the Figure. Inverters for the complemented variables are not shown.



Solution: The output expression is:

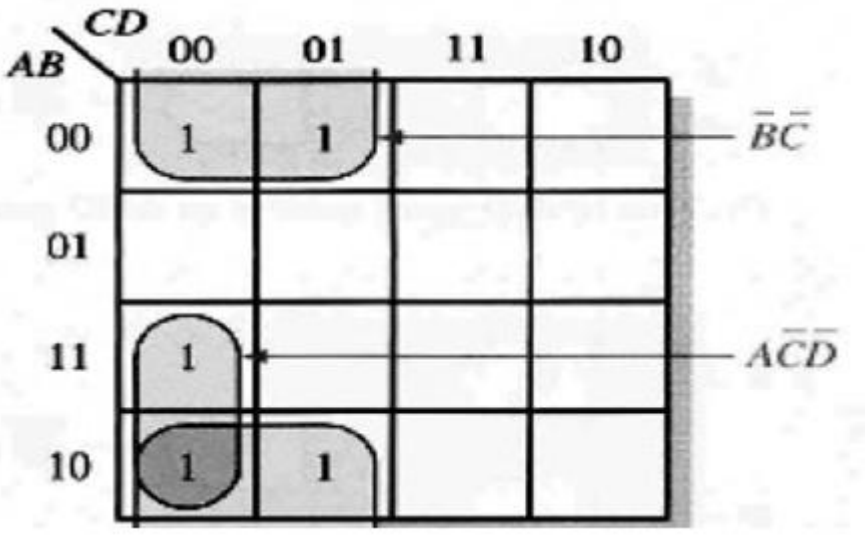
$$X = A\,\bar{B}\bar{C} + A\,B\,\bar{C}\,\bar{D} + \bar{A}\,\bar{B}\,\bar{C}\,D + \bar{A}\,\bar{B}\,\bar{C}\,\bar{D}$$

- Expanding the first term to include the missing variables D and $\bar{D}$ (to express in standard form).

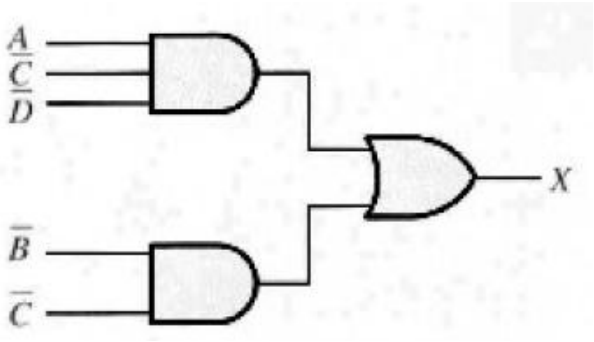
$$X = A\overline{B}\overline{C}(D + \overline{D}) + AB\overline{C}\overline{D} + \overline{A}BC\overline{D} + \overline{A}B\overline{C}D$$
$$X = A\overline{B}\overline{C}D + A\overline{B}\overline{C}\overline{D} + AB\overline{C}\overline{D} + \overline{A}BC\overline{D} + \overline{A}B\overline{C}D$$

- This expanded SOP expression is mapped and simplified on the Karnaugh map in the Figure.

$$X = A\,\bar{C}\,\bar{D} + \bar{B}\,\bar{C}$$



- The simplified implementation is shown in the Figure. Inverters are not shown.

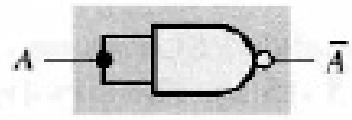


5.3. The Universal Property of NAND and NOR Gates:

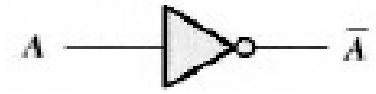
- Up to this point, you have studied combinational circuits implemented with AND gates, OR gates, and inverters.
- In this section, the universal property of the NAND gate and the NOR gate is discussed.
- The universality of the NAND gate means that it can be used as an inverter and that combinations of NAND gates can be used to implement the AND, OR, and NOR operations.
- Similarly, the NOR gate can be used to implement the inverter (NOT), AND, OR, and NAND operations.

5.3.1. The NAND Gate as a Universal Logic Element.

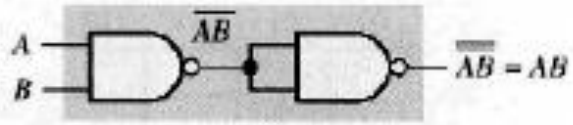
- The NAND gate is a universal gate because it can be used to produce the NOT, the AND, the OR, and the NOR functions.
- An inverter can be made from a NAND gate by connecting all of the inputs together and creating, in effect, a single input, as shown in the Figure for a 2-input gate.



(a) One NAND gate used as an inverter



- An AND function can be generated by the use of NAND gates alone, as shown in the Figure.

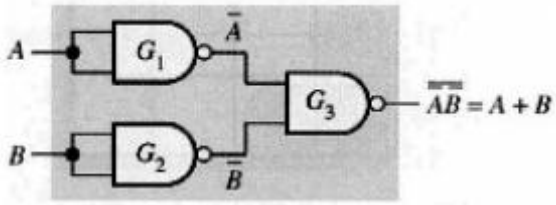


(b) Two NAND gates used as an AND gate

- A NAND gate is used to invert (complement) a NAND output to form the AND function, as indicated in the following equation:

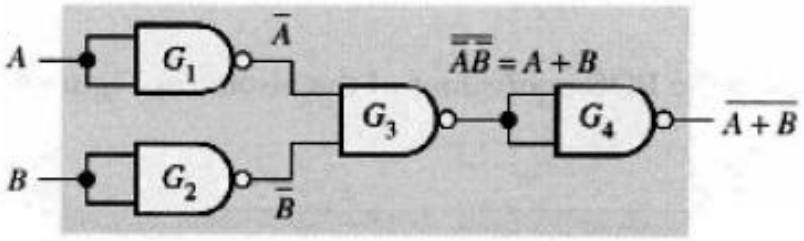
$$X = \overline{\overline{AB}} = AB$$

- An OR function can be produced with only NAND gates, as illustrated in the Figure.

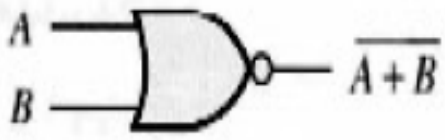


(c) Three NAND gates used as an OR gate

- NAND gates G1 and G2 are used to invert the two input variables before they are applied to NAND gate G3.
- The final OR output is derived as follows by application of DeMorgan's theorem: $X = \overline{\bar{A}\,\bar{B}} = A + B$
- Finally, a NOR function is produced as shown in the Figure.



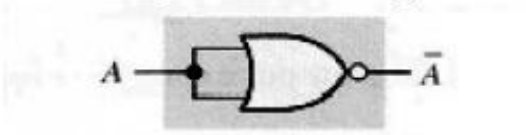
(d) Four NAND gates used as a NOR gate



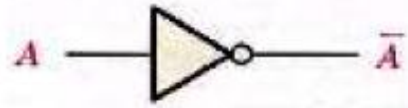
- NAND gate G4, is used as an inverter connected to the circuit shown in the above Figure to produce the NOR operation $\overline{A + B}$.

5.3.2. The NOR Gate as a Universal Logic Element

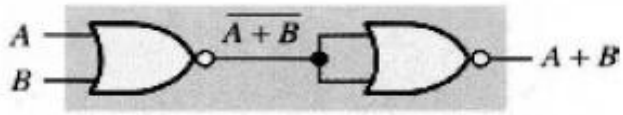
- Like the NAND gate, the NOR gate can be used to produce the NOT, AND. OR and NAND functions.
- A NOT circuit, or inverter, can be made from a NOR gate by connecting all of the inputs together to effectively create a single input, as shown in the Figure with a 2-input NOR.



(a) One NOR gate used as an inverter

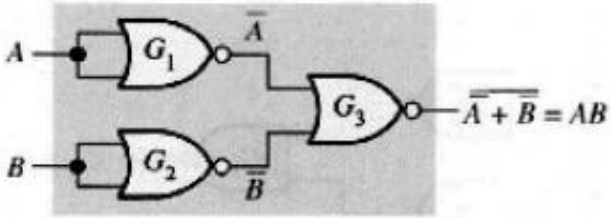


- Also, an OR gate can be produced from NOR gates, as illustrated in the Figure.



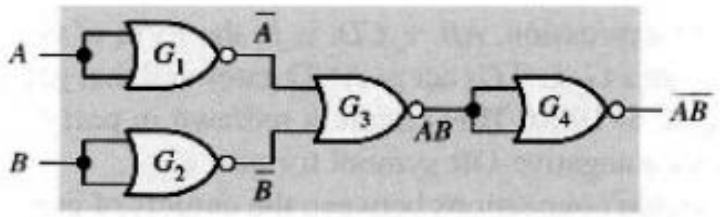
(b) Two NOR gates used as an OR gate

- An AND gate can be constructed by the use of NOR gates, as shown in the Figure.

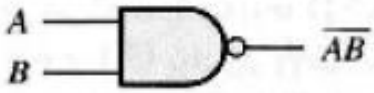


(c) Three NOR gates used as an AND gate

- In this case the NOR gates G1 and G2 are used as inverters, and the final output is derived by the use of DeMorgan's theorem as follows: $X = \overline{\bar{A} + \bar{B}} = A\,B$.
- The Figure shows how NOR gates are used to form a NAND function.



(d) Four NOR gates used as a NAND gate



5.4. Combinational Logic Using NAND And NOR Gates.

- In this section, you will see how NAND and NOR gates can be used to implement a logic function.

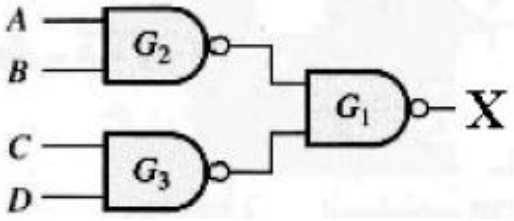
5.4.1. NAND Logic.

- As you have learned, a NAND gate can function as either a NAND or a negative-OR because, by DeMorgan's theorem:

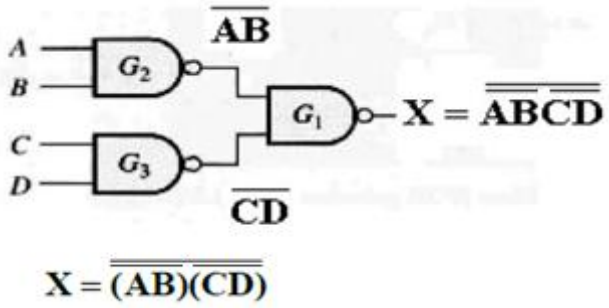
$$\overline{AB} = \bar{A} + \bar{B}$$

NAND — negative-OR

- Consider the NAND logic in the Figure.



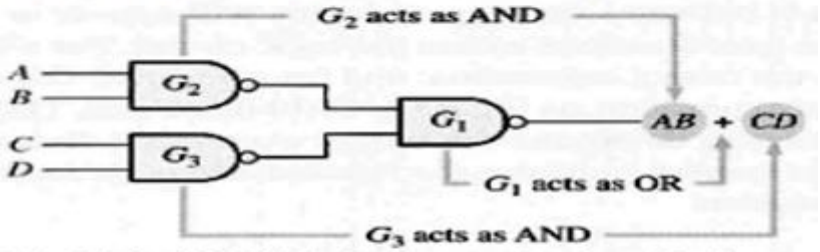
- **Step 1:** Determine the output expression of the logic circuit as:



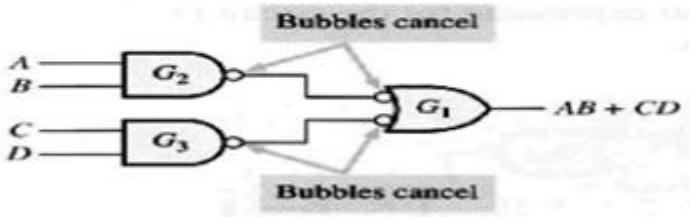
$$X = \overline{(\overline{AB})(\overline{CD})}$$

- Step 2: Simplify the output expression as follows:

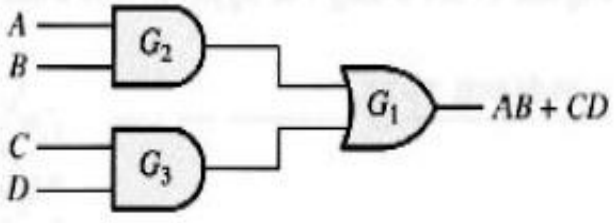
$$X = \overline{(\overline{AB})(\overline{CD})}$$
$$X = \overline{(\overline{A} + \overline{B})} + \overline{(\overline{C} + \overline{D})}$$
$$X = (\overline{\overline{A} + \overline{B}}) \times (\overline{\overline{C} + \overline{D}})$$
$$X = \overline{\overline{A}\,\overline{B}} + \overline{\overline{C}\,\overline{D}}$$
$$X = AB + CD \qquad \text{SOP Form}$$



(a) Original NAND logic diagram showing effective gate operation relative to the output expression



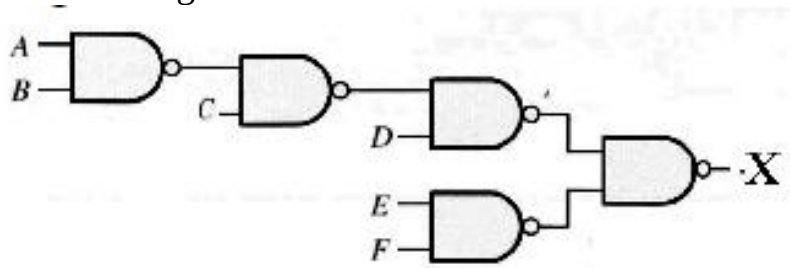
(b) Equivalent NAND/Negative-OR logic diagram



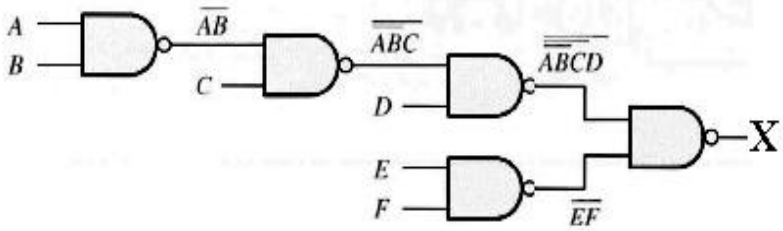
(c) AND-OR equivalent

5.4.2. NAND Logic Diagrams Using Dual Symbols.

- All logic diagrams using NAND gates should be drawn with each gate represented by either a NAND symbol or the equivalent negative-OR symbol to reflect the operation of the gate within the logic circuit.
- The NAND symbol and the negative-OR symbol are called dual symbol.
- When drawing a NAND logic diagram, always use the gate symbols in such a way that every connection between a gate output and a gate input is either bubble-to-bubble or nonbubble-to-nonbubble.
- A bubble output should not be connected to a nonbubble input or vice versa in a logic diagram.
- The Figure shows an arrangement of gates to illustrate the procedure of using the appropriate dual symbols for a NAND circuit with several gate levels.



- **Step 1:** Determine the output expression of the logic circuit.

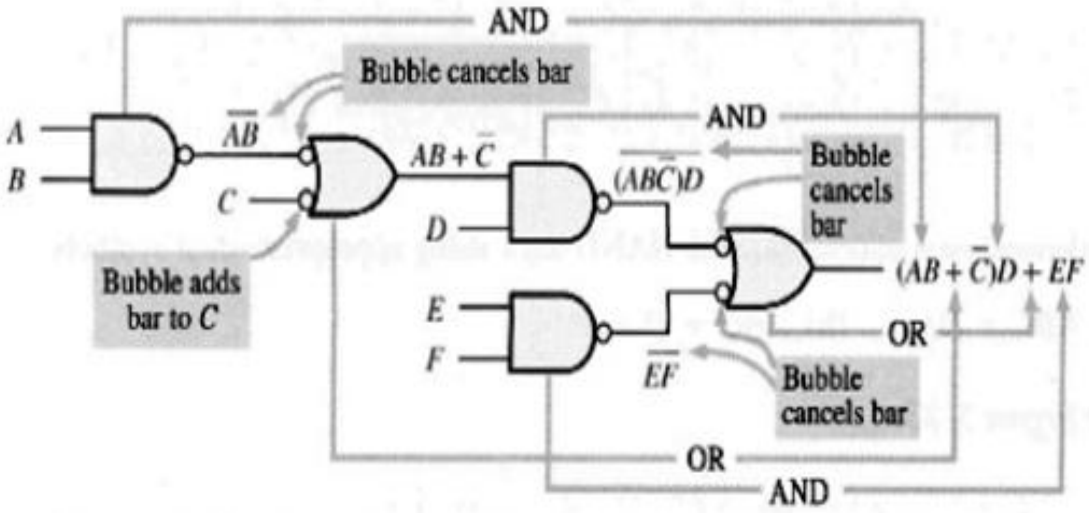


$$X = \overline{(\overline{ABCD})(\overline{EF})}$$

- **Step 2:** Simplify the output expression.

$$X = \overline{(\overline{ABCD})} + \overline{(\overline{EF})}$$
$$X = \overline{\overline{ABCD}} + EF$$
$$X = (\overline{\overline{AB} + \overline{C}})D + EF$$
$$X = (AB + \overline{C})D + EF$$

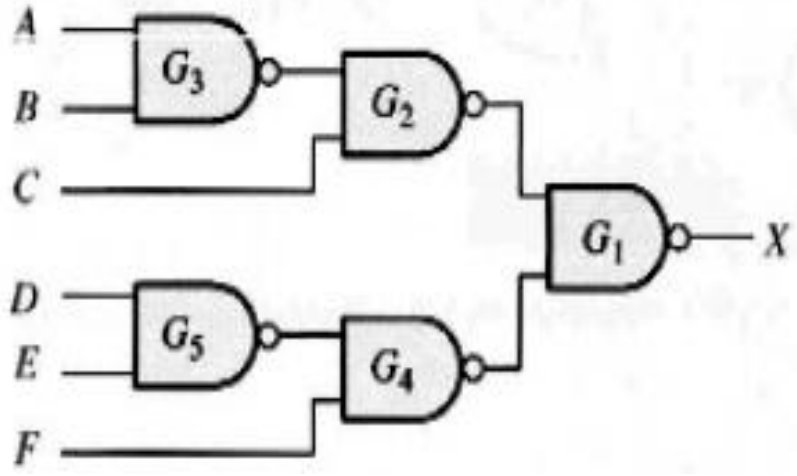
- Although using all NAND symbols as in the above Figure is correct, the following diagram is much easier to "read" and is the preferred method.



Output expression can be obtained directly from the function of each gate symbol in the diagram.

- As shown in Figure above, the output gate is represented with a negative-OR symbol.
- Then the NAND symbol is used for the level of gates right before the output gate and the symbols for successive levels of gates are alternated as you move away from the output.

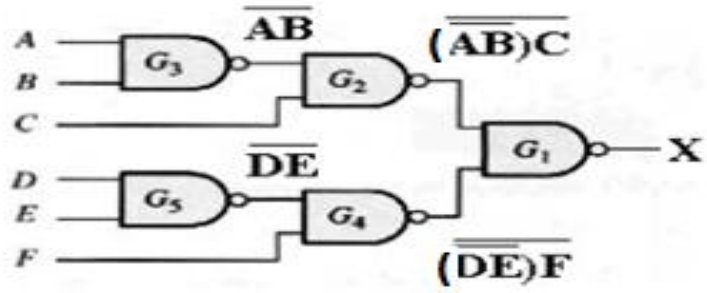
Example (7): (a)-Develop the output expression the combinational logic circuit shown in the Figure.



(b) Redraw the logic diagram and in the Figure using the appropriate dual symbols.

Solution:

(a) The output expression of the circuit is:



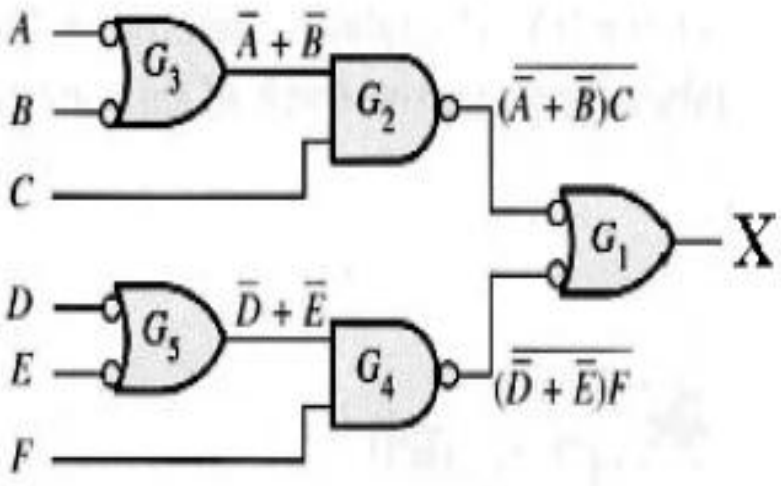
$$X = \overline{((\overline{AB})C) \times ((\overline{DE}) \times F)}$$

$$X = \overline{((\overline{AB})C)} + \overline{((\overline{DE}) \times F)}$$

$$X = (\overline{AB})C + (\overline{DE}) \times F$$

$$X = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$$

(b) Redraw the logic diagram in the above Figure with the use of equivalent negative-OR symbols as shown in the Figure.



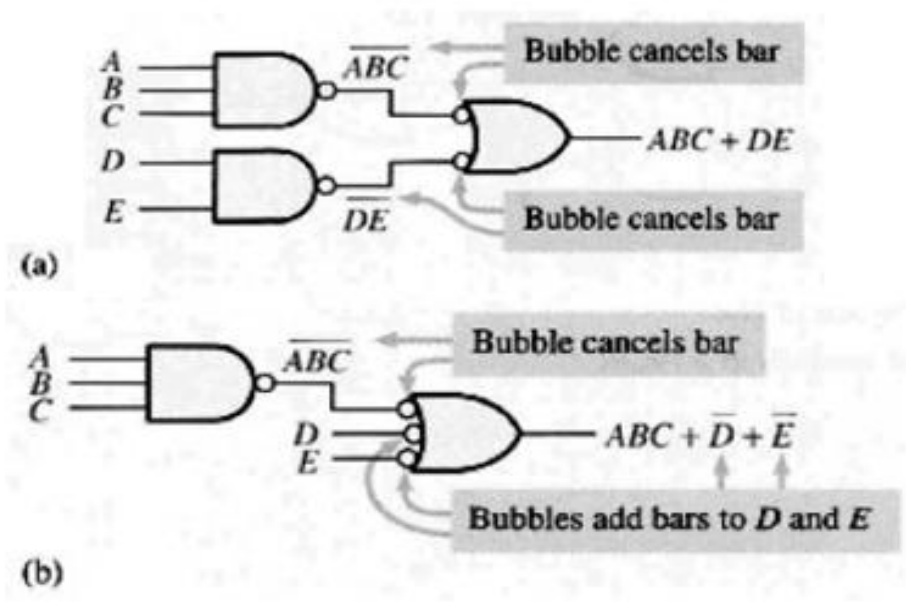
- Writing the expression for X directly from the indicated logic operation of each gate gives:

$$X = (\bar{A} + \bar{B})C + (\bar{D} + \bar{E})F$$

Example (8): Implement each expression with NAND logic using appropriate dual symbols:

(a) ABC+DE (b) $ABC + \bar{D} + \bar{E}$

Solution:



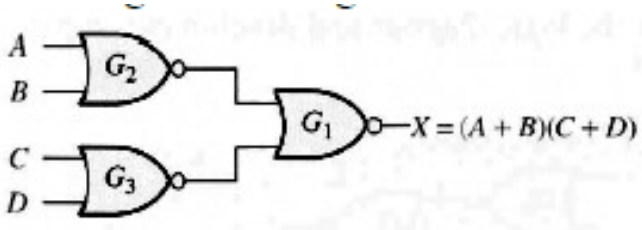
5.4.3. NOR Logic.

- A NOR gate can function as either a NOR or a negative-AND, as shown by DeMorgan's theorem.

$$\overline{A + B} = \overline{A}\,\overline{B}$$

NOR ↑ ↑ negative-AND

- Consider the NOR logic in the Figure.

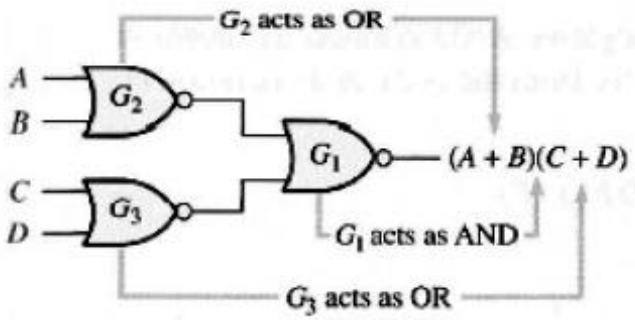


- The output expression is developed as follows:

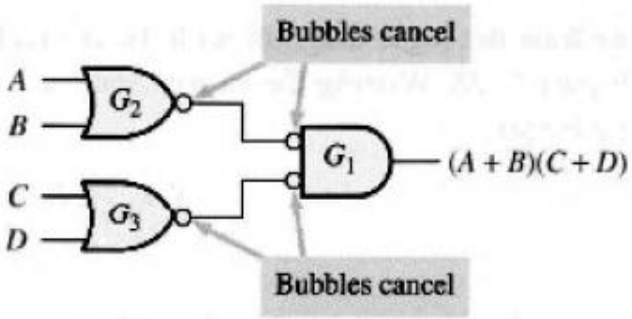
$$X = \overline{(\overline{A + B})} + \overline{(\overline{C + D})}$$
$$X = \overline{\overline{(A + B)}} \times \overline{\overline{(C + D)}}$$
$$X = (A + B) \times (C + D) \qquad \text{POS Form}$$

- As you can see in the above Figure, the output expression (A + B) x (C + D) consists of two OR terms ANDed together.

- This shows that gates G2 and G3 act as OR gates and gate G1 acts as an AND gate, as illustrated in the following Figure.

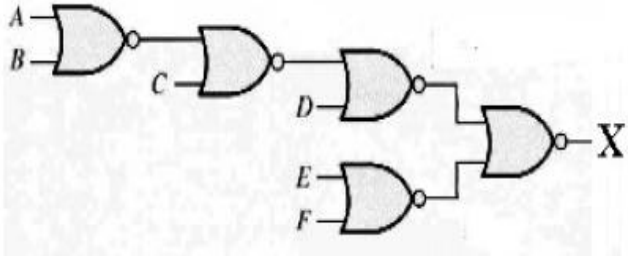


- This circuit is redrawn in this Figure with a negative-AND symbol for gate G1.

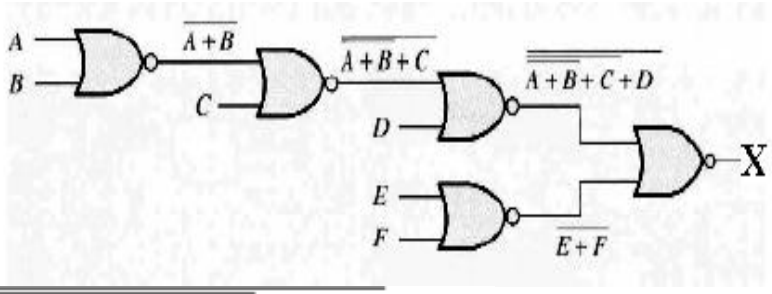


5.4.4. NOR Logic Diagram Using Dual Symbol.

- As with NAND logic, the purpose for using **the dual symbols** is to make the logic diagram easier to read and analyze, as illustrated in the NOR logic circuit in the Figure.



- Develop the output expression X



$$X = \overline{\overline{((A+B)+C+D)}+\overline{E+F}}$$

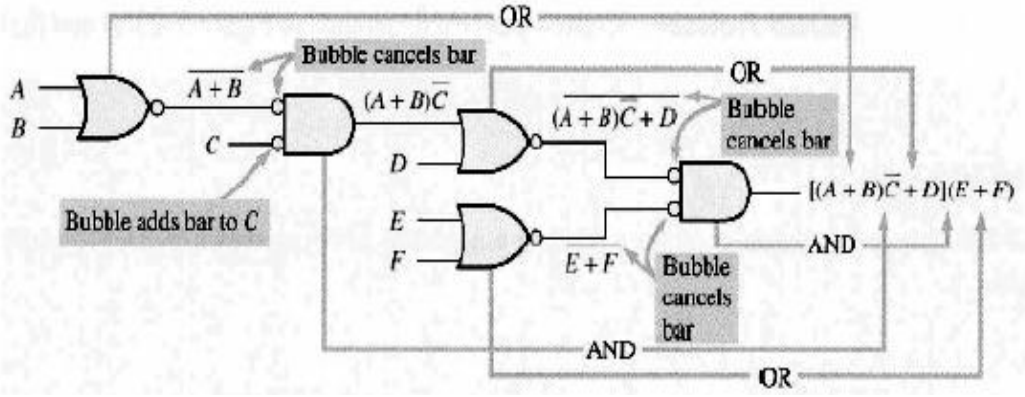
$$X = \overline{\overline{((A+B)+C+D)}} \times \overline{\overline{(E+F)}}$$

$$X = \overline{\overline{((A+B)+C+D)}} \times (E+F)$$

$$X = \overline{\overline{((A+B)} \times \overline{C}+D)} \times (E+F)$$

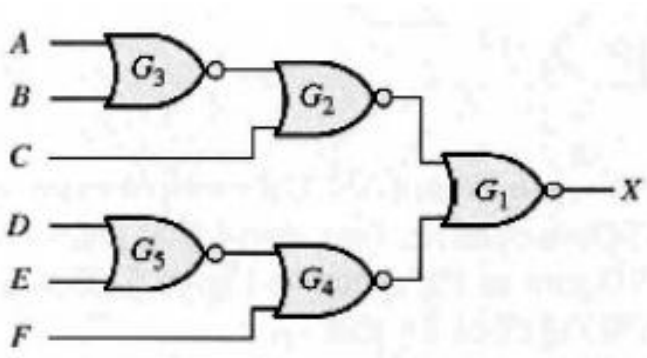
$$X = ((A+B) \times \overline{C}+D) \times (E+F)$$

- When the circuit above is redrawn with dual symbols in the following Figure, notice that all output-to-input connections between gates are bubble-to-bubble or nonbubble-to-nonbubble.



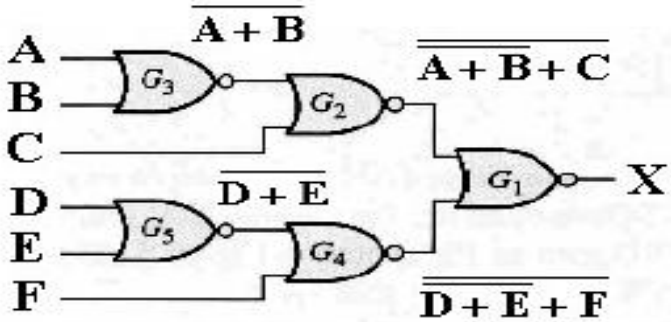
- Again, you can see that the shape of each gate symbol indicates the type of term (AND or OR) that it produces in the output expression, thus making the output expression easier to determine and the logic diagram easier to analyze.

Example (9): Using appropriate dual symbol redraw the logic diagram and develop the output expression for the circuit in the Figure.



Solution:

a. Determine the output expression of the logic circuit as follows:



$$X = \overline{\overline{(A + B + C)} + \overline{(D + E + F)}}$$

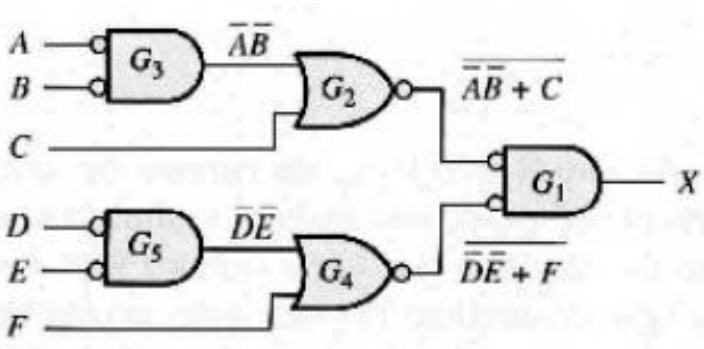
b. Simplify the output expression as follows:

$$X = \overline{\overline{(A + B + C)}} \times \overline{\overline{(D + E + F)}}$$

$$X = (A + B + C) \times (D + E + F)$$

$$X = (\bar{A}\bar{B} + C) \times (\bar{D}\bar{E} + F)$$

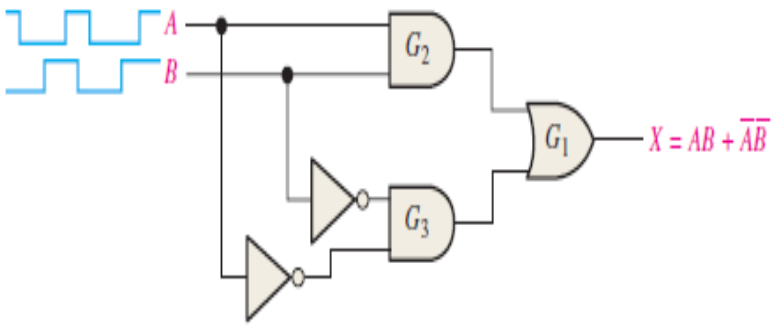
- Redraw the logic diagram with the equivalent negative-AND symbols as shown in the Figure.



- Writing the expression for X directly from the indicated operation of each gate X = ($\bar{A}$ $\bar{B}$ + C) x ($\bar{D}$ $\bar{E}$ + F)

Example (10):

Draw the timing diagram for the circuit in Figure below showing the outputs of *G*1, *G*2, and *G*3 with the input waveforms, *A*, and *B*, as indicated.



Solution

When both inputs are HIGH or when both inputs are LOW, the output *X* is HIGH as shown below. Notice that this is an exclusive-NOR circuit. The intermediate outputs of gates *G*2 and *G*3 are also shown below.