Lecture Five

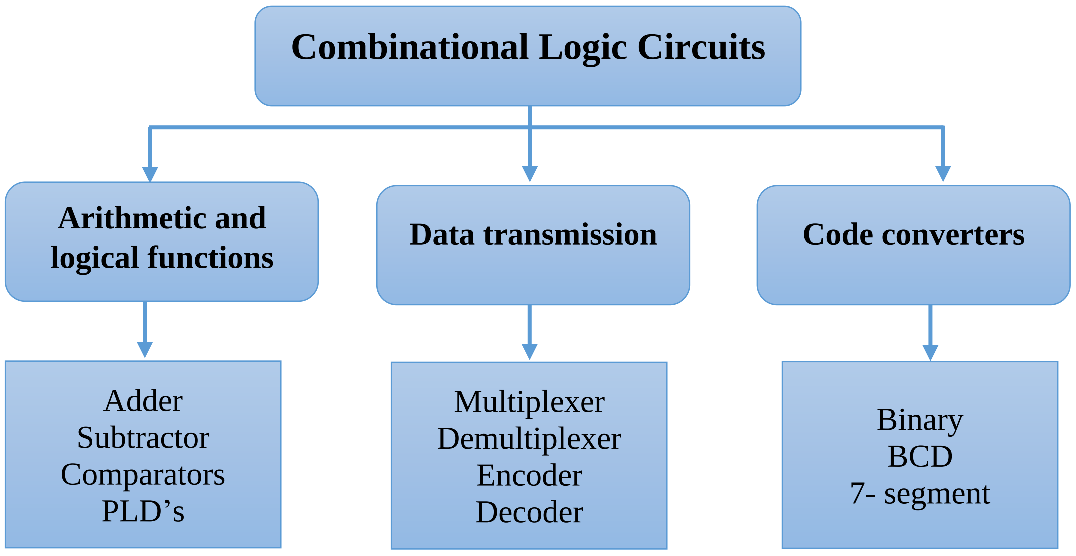
Combinational Logic Analysis

Outline

1- Basic Combinational Logic Circuits
2- Implementing Combinational Logic
3- The Universal Property of NAND and NOR Gates
4- Combinational Logic Using NAND and NOR Gates
5- Pulse Waveform Operation

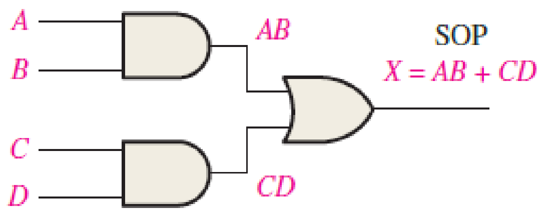
Basic Combinational Logic Circuits

Basic combinational logic circuits are digital circuits in which the output depends only on the present input values, without memory or feedback loops. These circuits perform logical operations such as AND, OR, NOT, XOR, etc., and are widely used in arithmetic and data processing applications.



AND-OR Logic

The Boolean expressions for the AND gate outputs and the resulting SOP expression for the output *X* are shown on the figure below. In general, an AND-OR circuit can have any number of AND gates, each with any number of inputs.

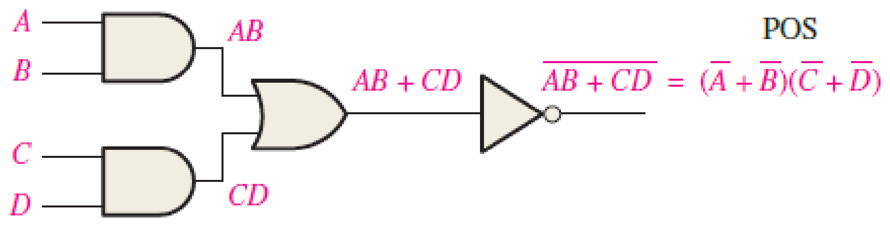


Input						Output
A	B	C	D	AB	CD	X
0	0	0	0	0	0	**0**
0	0	0	1	0	0	**0**
0	0	1	0	0	0	**0**
0	0	1	1	0	1	**1**
0	1	0	0	0	0	**0**
0	1	0	1	0	0	**0**
0	1	1	0	0	0	**0**
0	1	1	1	0	1	**1**
1	0	0	0	0	0	**0**
1	0	0	1	0	0	**0**
1	0	1	0	0	0	**0**
1	0	1	1	0	1	**1**
1	1	0	0	1	0	**1**
1	1	0	1	1	0	**1**
1	1	1	0	1	0	**1**
1	1	1	1	1	1	**1**

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).

AND-OR-Invert Logic

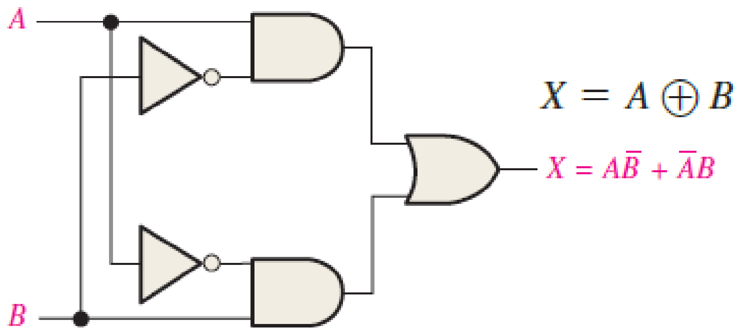
When the output of an AND-OR circuit is complemented (inverted), it results in an AND-OR-Invert circuit. Recall that AND-OR logic directly implements SOP expressions. POS expressions can be implemented with AND-OR-Invert logic.


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

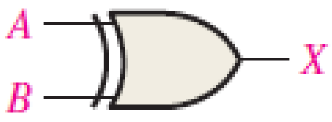
$$X = (\overline{A} + \overline{B})(\overline{C} + \overline{D}) = (\overline{AB})(\overline{CD}) = \overline{\overline{(\overline{AB})(\overline{CD})}} = \overline{\overline{AB}} + \overline{\overline{CD}} = \overline{AB + CD}$$

Exclusive-OR Logic

The exclusive-OR gate is a combination of two AND gates, one OR gate, and two inverters, as shown in Figure below



Input		Output
B	A	Y
0	0	0
0	1	1
1	0	1
1	1	0

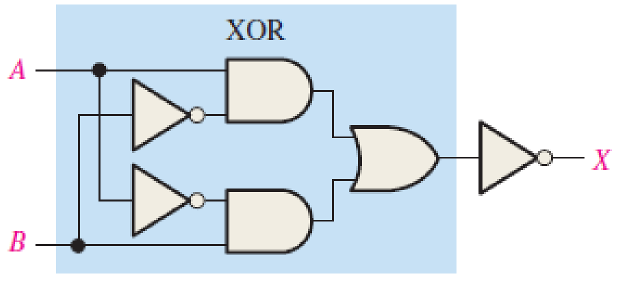


The output is HIGH only when the two inputs are at opposite levels.

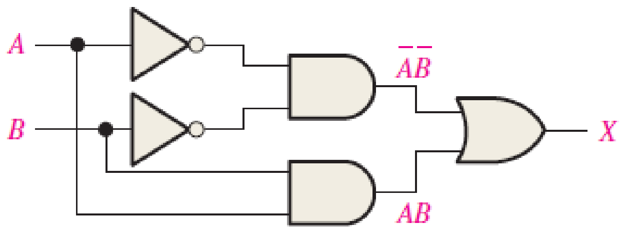
Exclusive-NOR Logic

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

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



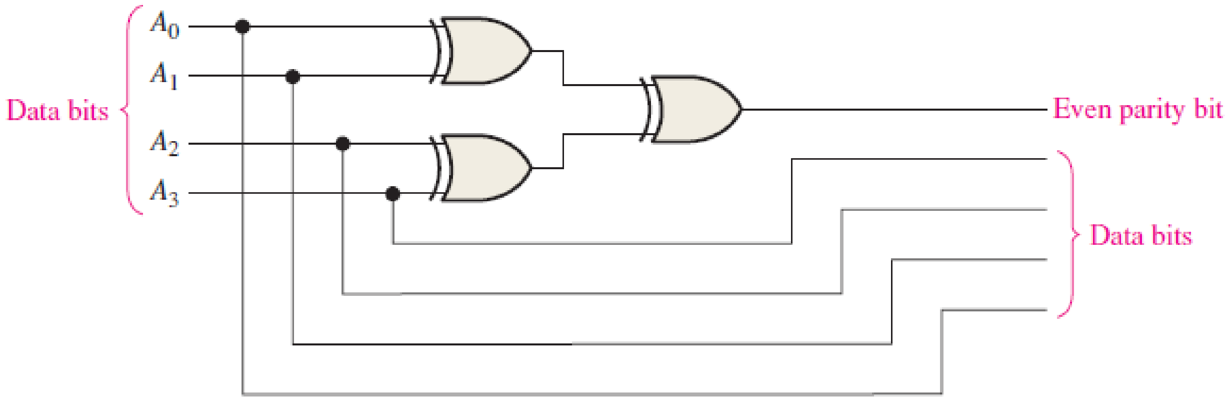
(a) $X = \overline{A\overline{B} + \overline{A}B}$



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

Example: Use exclusive-OR gates to implement **an even-parity code generator** for an original 4-bit code.

Solution



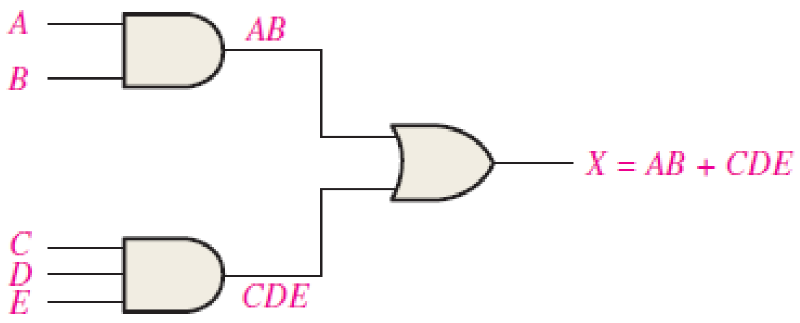
Evan parity generator

Implementing Combinational Logic

In this section, examples are used to illustrate how to implement a logic circuit from a Boolean expression or a truth table.

From a Boolean Expression to a Logic Circuit

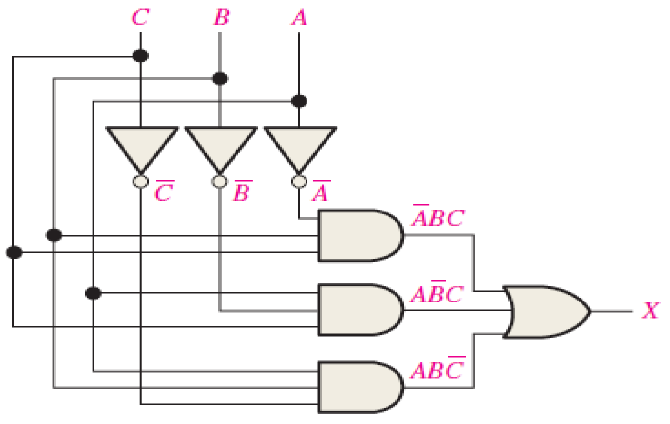
Let's examine the following Boolean expression: $X = AB + CDE$



From a Truth Table to a Logic Circuit

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

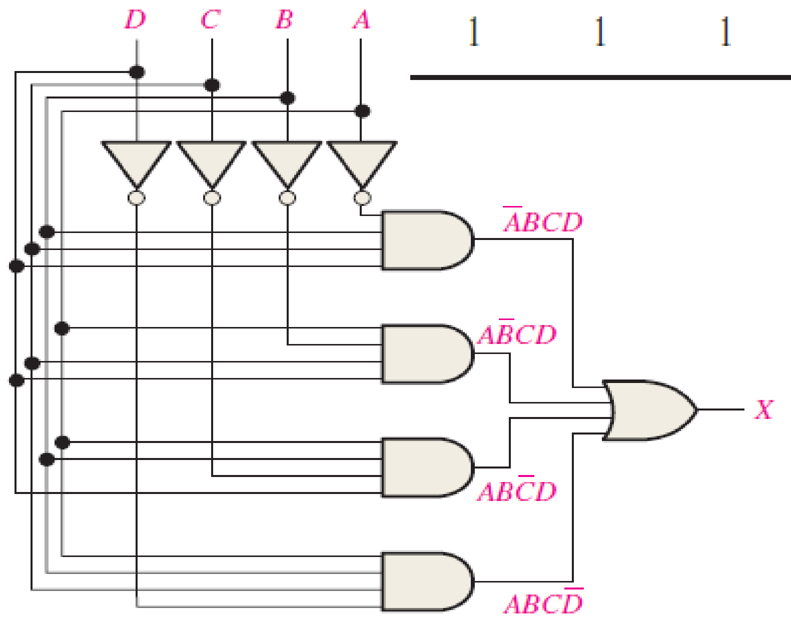
Inputs			Output	
A	*B*	*C*	*X*	Product Term
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	0	



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

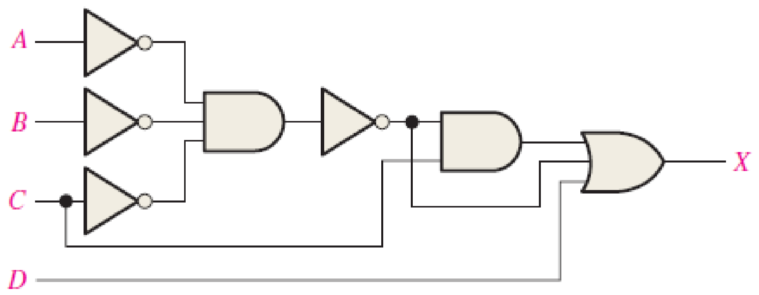
Solution

A	B	C	D	Product Term
0	1	1	1	$\overline{A}BCD$
1	0	1	1	$A\overline{B}CD$
1	1	0	1	$AB\overline{C}D$
1	1	1	0	$ABC\overline{D}$



$$\overline{A}BCD + A\overline{B}CD + AB\overline{C}D + ABC\overline{D}$$

Example: Reduce the combinational logic circuit in Figure below to a minimum form.



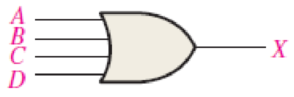
Solution

The expression for the output of the circuit is

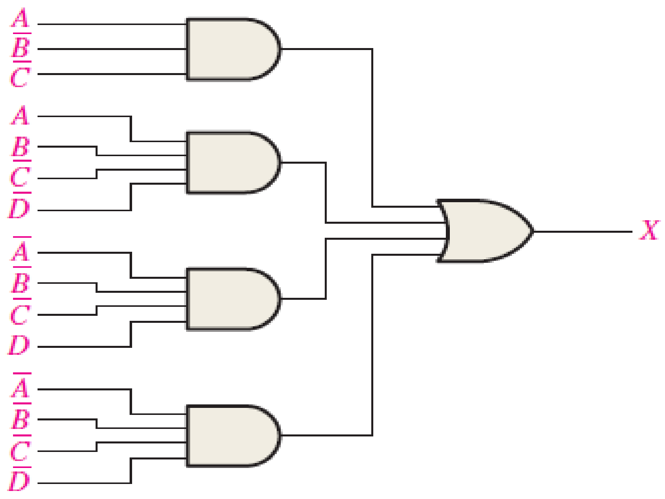
$$X = (\overline{\overline{A}\,\overline{B}\,\overline{C}})C + \overline{\overline{A}\,\overline{B}\,\overline{C}} + D$$

Applying DeMorgan's theorem and Boolean algebra,

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



Example 2: Minimize the combinational logic circuit in Figure below. Inverters for the complemented variables are not shown

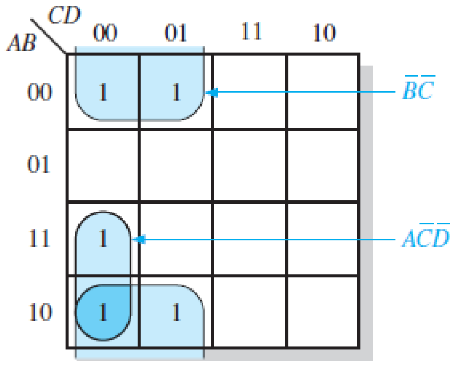


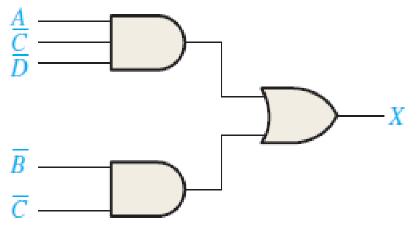
Solution

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

Expanding the first term to include the missing variables D and $\overline{D}$,

$$\begin{aligned} X &= A\overline{B}\,\overline{C}(D + \overline{D}) + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} \\ &= A\overline{B}\,\overline{C}D + A\overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} \end{aligned}$$

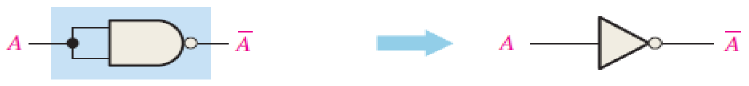




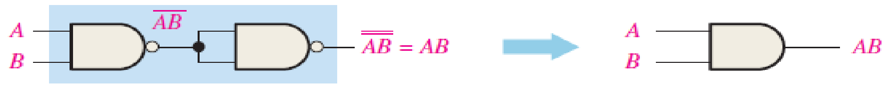
The Universal Property of NAND and NOR Gates

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.

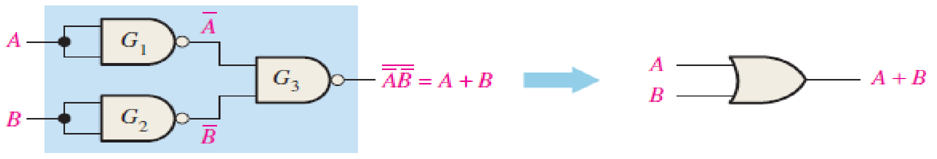
The NAND gate is a **universal gate** because it can be used to produce the NOT, the AND, the OR, and the NOR functions. An inverter can be made from a NAND gate by connecting all of the inputs together and creating, in effect, a single input



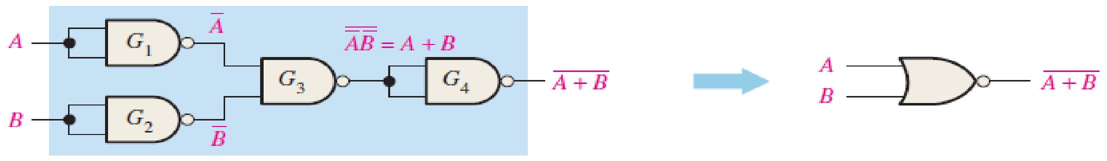
(a) One NAND gate used as an inverter



(b) Two NAND gates used as an AND gate

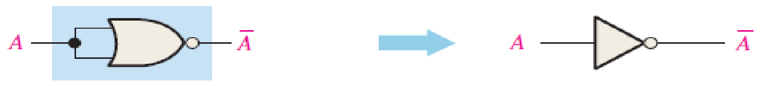


(c) Three NAND gates used as an OR gate

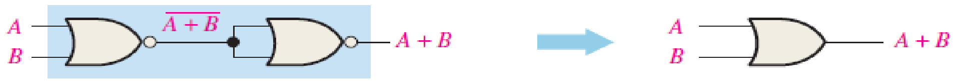


(d) Four NAND gates used as a NOR gate

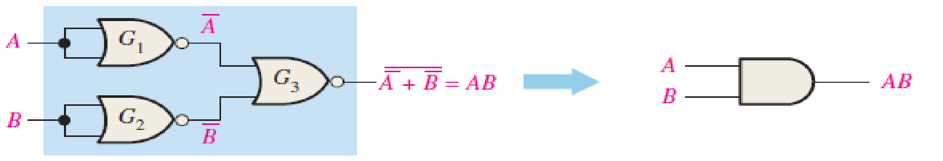
The NOR Gate as a Universal Logic Element



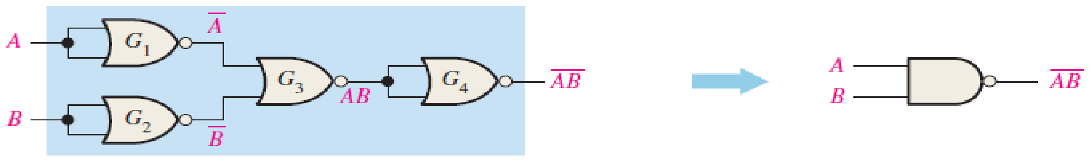
(a) One NOR gate used as an inverter



(b) Two NOR gates used as an OR gate



(c) Three NOR gates used as an AND gate



(d) Four NOR gates used as a NAND gate

Combinational Logic Using NAND and NOR Gates

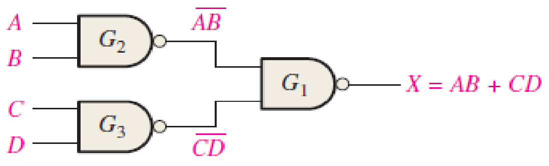
NAND Logic

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

$$\underset{\text{NAND}}{\overline{AB}} = \underset{\text{negative-OR}}{\overline{A} + \overline{B}}$$

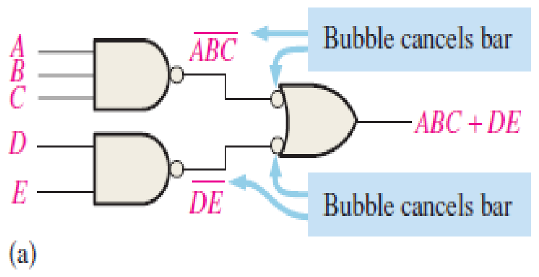
Consider the NAND logic in Figure 5–20. The output expression is developed in the following steps:

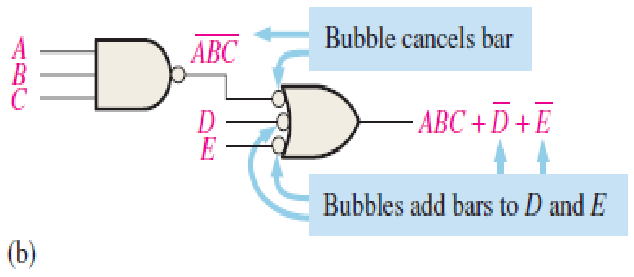
$$\begin{aligned} X &= \overline{(\overline{AB})(\overline{CD})} \\ &= \overline{(\overline{A} + \overline{B})(\overline{C} + \overline{D})} \\ &= (\overline{\overline{A} + \overline{B}}) + (\overline{\overline{C} + \overline{D}}) \\ &= \overline{\overline{A}}\,\overline{\overline{B}} + \overline{\overline{C}}\,\overline{\overline{D}} \\ &= AB + CD \end{aligned}$$



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

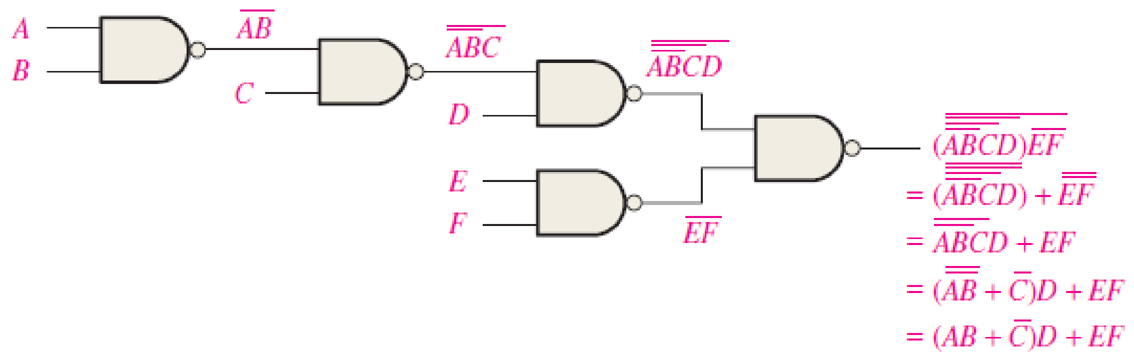
(a) $ABC + DE$ (b) $ABC + \overline{D} + \overline{E}$



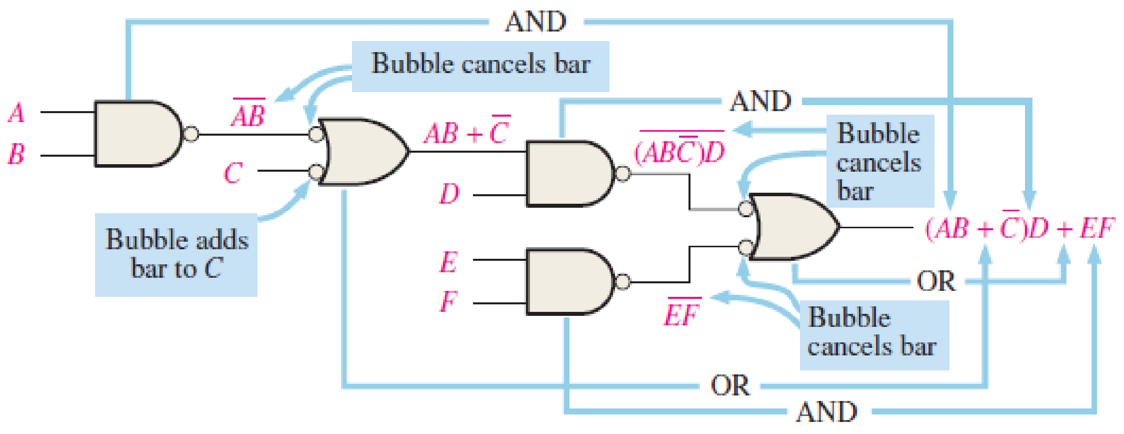

A
B
C
D
E
$\overline{ABC}$
Bubble cancels bar
$ABC + \overline{D} + \overline{E}$
Bubbles add bars to D and E
(b)

NAND Logic Diagrams Using Dual Symbols

The NAND symbol and the **negative-OR** symbol are called ***dual symbols***

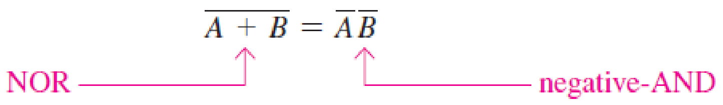

A
B
$\overline{AB}$
C
$\overline{\overline{AB}C}$
D
$\overline{\overline{\overline{AB}C}D}$
E
F
$\overline{EF}$
$\overline{(\overline{\overline{AB}C}D)\overline{EF}}$
$= (\overline{\overline{\overline{\overline{AB}C}D}}) + \overline{\overline{EF}}$
$= \overline{\overline{AB}C}D + EF$
$= (\overline{\overline{AB}}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

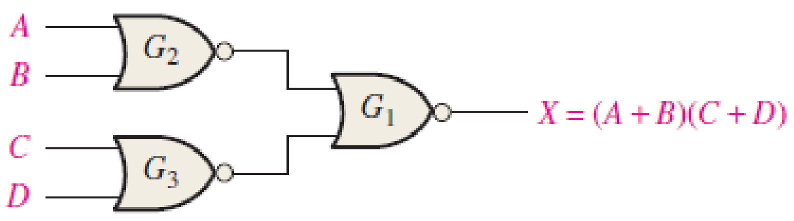
Use of the appropriate dual symbols in a NAND logic



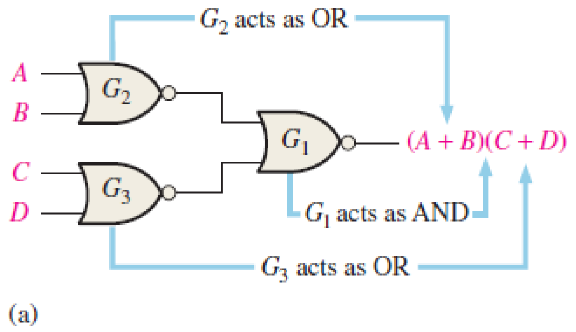
NOR Logic

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

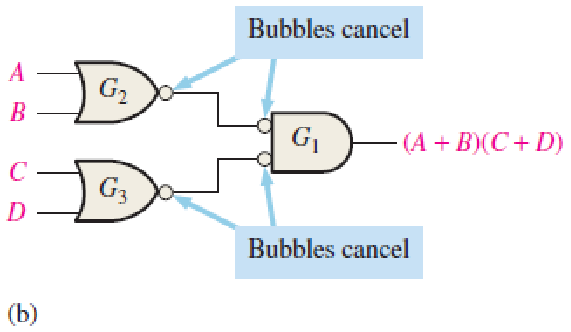




$$X = \overline{\overline{A + B} + \overline{C + D}} = (\overline{\overline{A + B}})(\overline{\overline{C + D}}) = (A + B)(C + D)$$

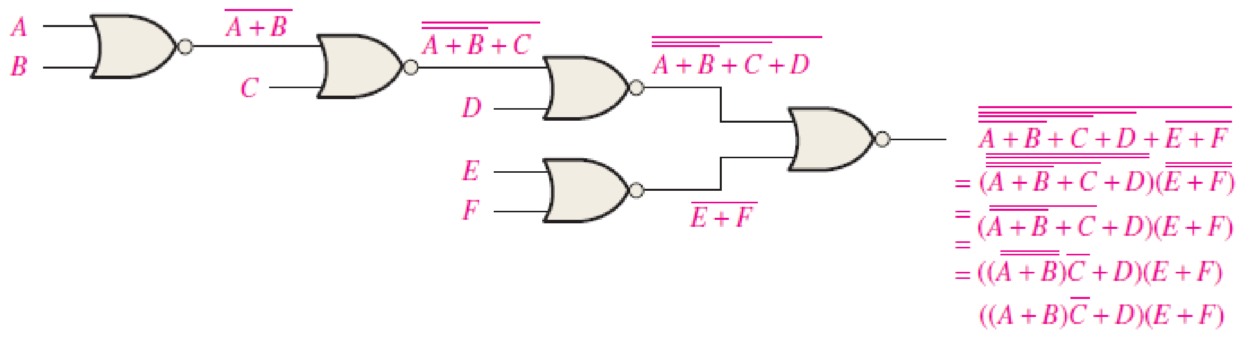


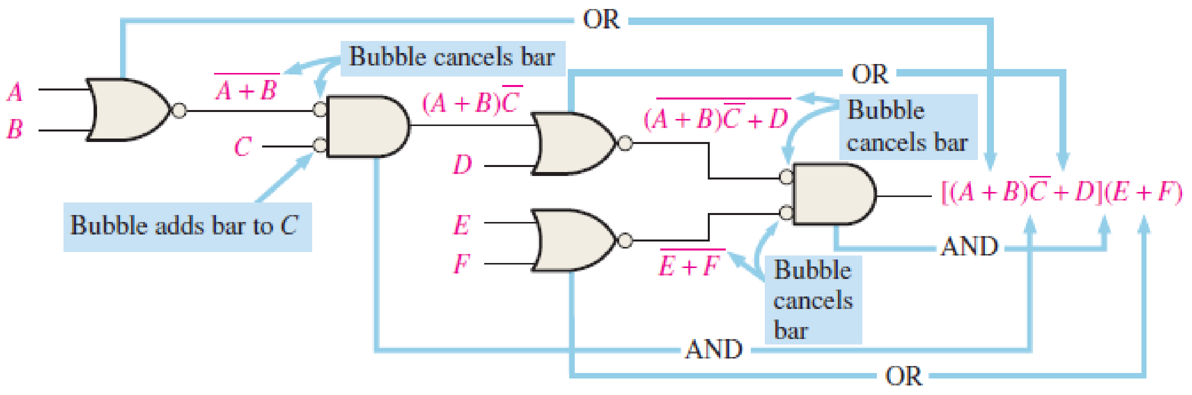
(a)



(b)

NOR Logic Diagram Using Dual Symbols


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



Pulse Waveform Operation

The following is a review of the operation of individual gates for use in analyzing combinational circuits with pulse waveform inputs:

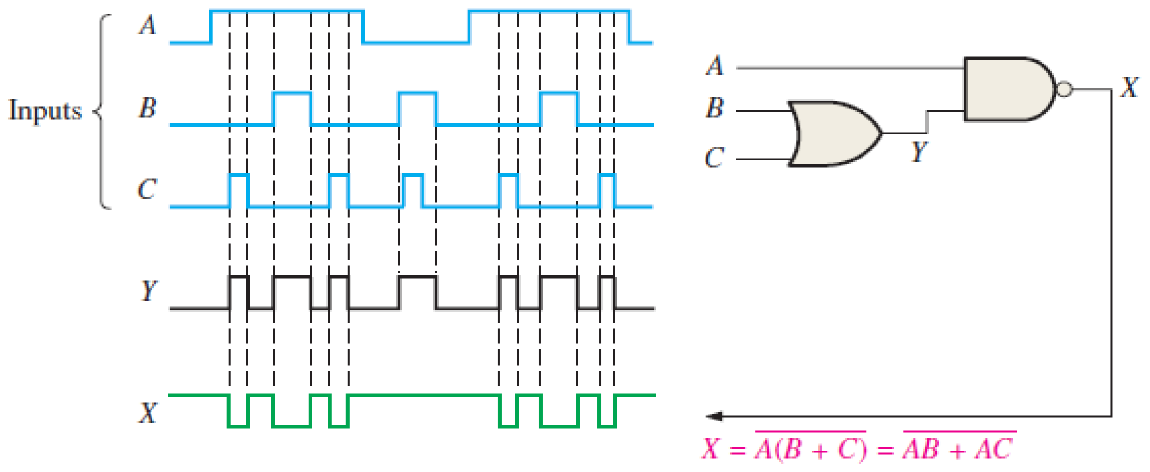
1. The output of an AND gate is HIGH only when all inputs are HIGH at the same time.

2. The output of an OR gate is HIGH only when at least one of its inputs is HIGH.

3. The output of a NAND gate is LOW only when all inputs are HIGH at the same time.

4. The output of a NOR gate is LOW only when at least one of its inputs is HIGH.

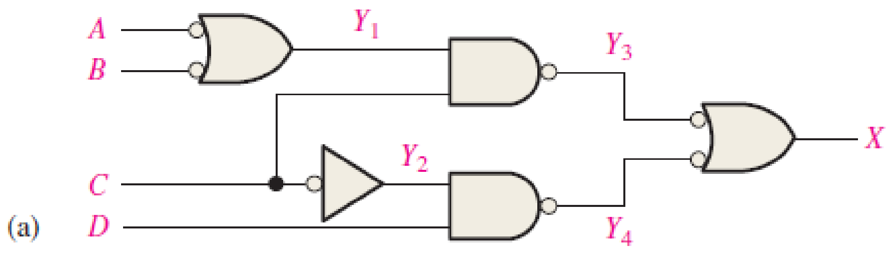
Example: Determine the final output waveform X for the circuit in Figure below, with input waveforms A, B, and C as shown.

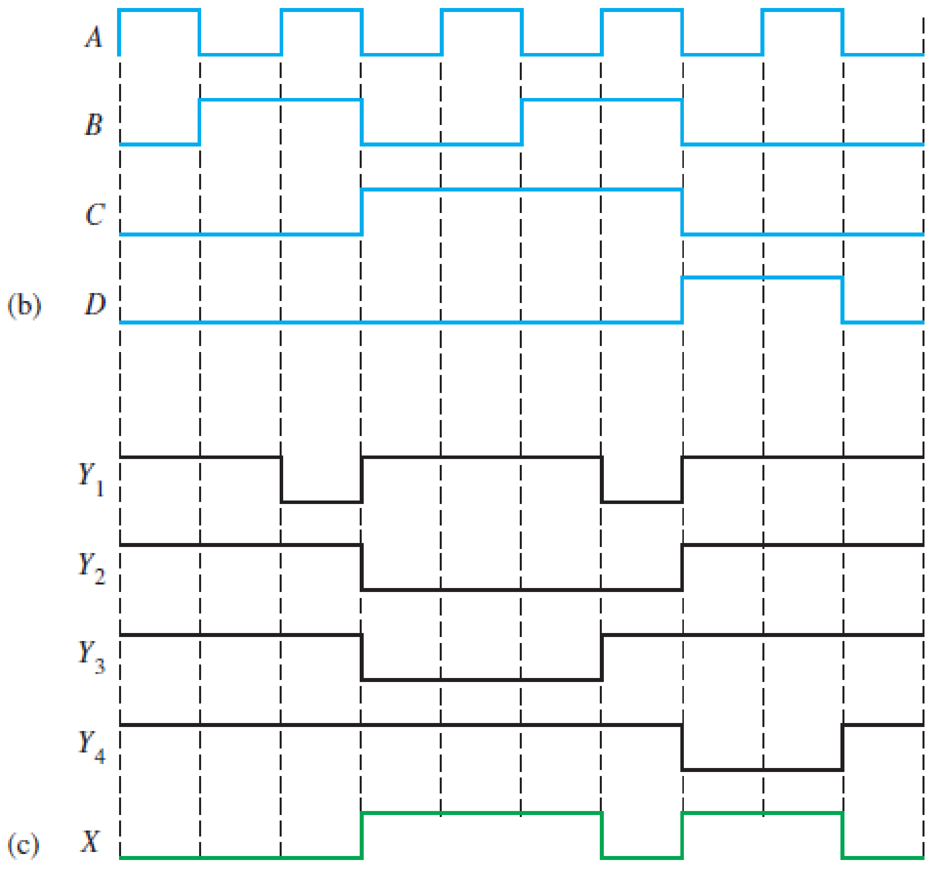


The output expression, $\overline{AB + AC}$, indicates that the output X is LOW when both A and B are HIGH or when both A and C are HIGH or when all inputs are HIGH.

Example: Determine the output waveform *X* for the logic circuit in Figure below by first finding the intermediate waveform at each of points *Y*1, *Y*2, *Y*3, and *Y*4.

Solution


(a)
A
B
C
D
Y1
Y2
Y3
Y4
X


(b)
A
B
C
D
(c)
Y1
Y2
Y3
Y4
X