Vietnam National University HCMC

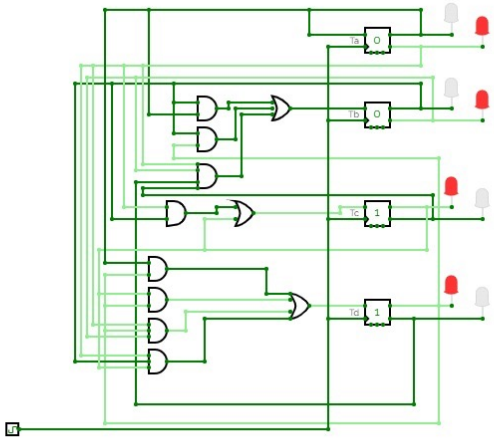
International University

School of Electrical Engineering

EE053IU

# Digital Logic Design

## Lecture 5: Combinational Logic Analysis

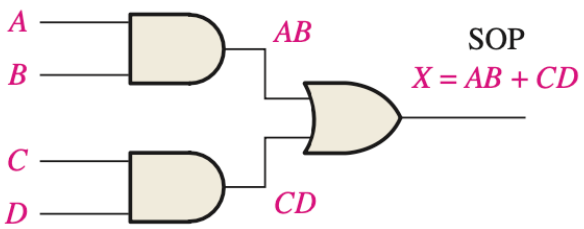


INSTRUCTOR: Dr. Vuong Quoc Bao

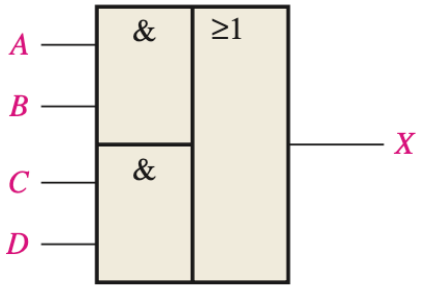
# 1. Basic Combinational Logic Circuits

## AND-OR Logic

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).




(a) Logic diagram (ANSI standard distinctive shape symbols)




(b) ANSI standard rectangular outline symbol

**FIGURE 5–1** An example of AND-OR logic. Open file F05-01 to verify the operation. *A Multisim tutorial is available on the website.*

**TABLE 5–1**

Truth table for the AND-OR logic in Figure 5–1.

| Inputs | | | | | | 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 |

# AND-OR-Invert Logic

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).

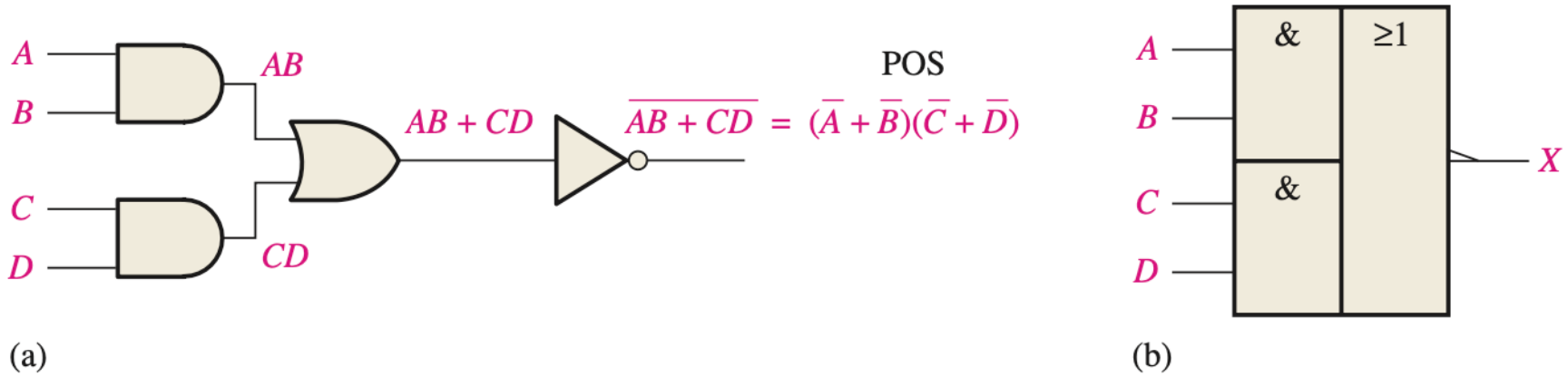
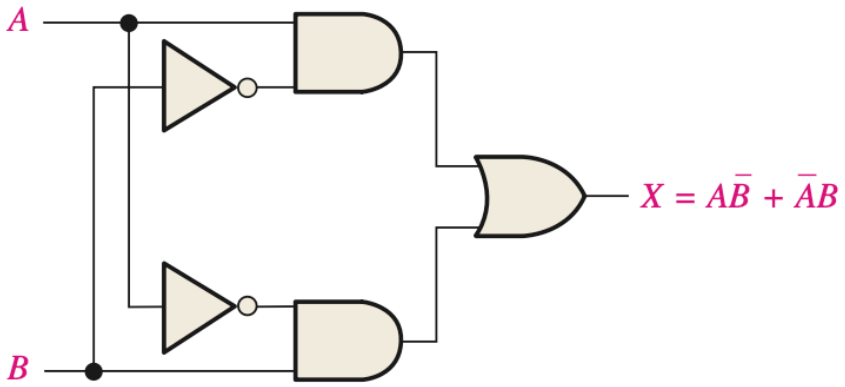



**FIGURE 5–3** An AND-OR-Invert circuit produces a POS output. Open file F05-03 to verify the operation.

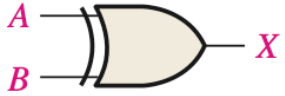
- A truth table can be developed from the AND-OR truth table in Table 5–1 by simply changing all 1s to 0s and all 0s to 1s in the output column.

# Exclusive-OR Logic

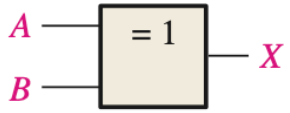



(a) Logic diagram

$$X = A \oplus B$$




(b) ANSI distinctive shape symbol




(c) ANSI rectangular outline symbol

**FIGURE 5–5** Exclusive-OR logic diagram and symbols. Open file F05-05 to verify the operation.

The XOR gate is actually a combination of other gates.

**TABLE 5–2**

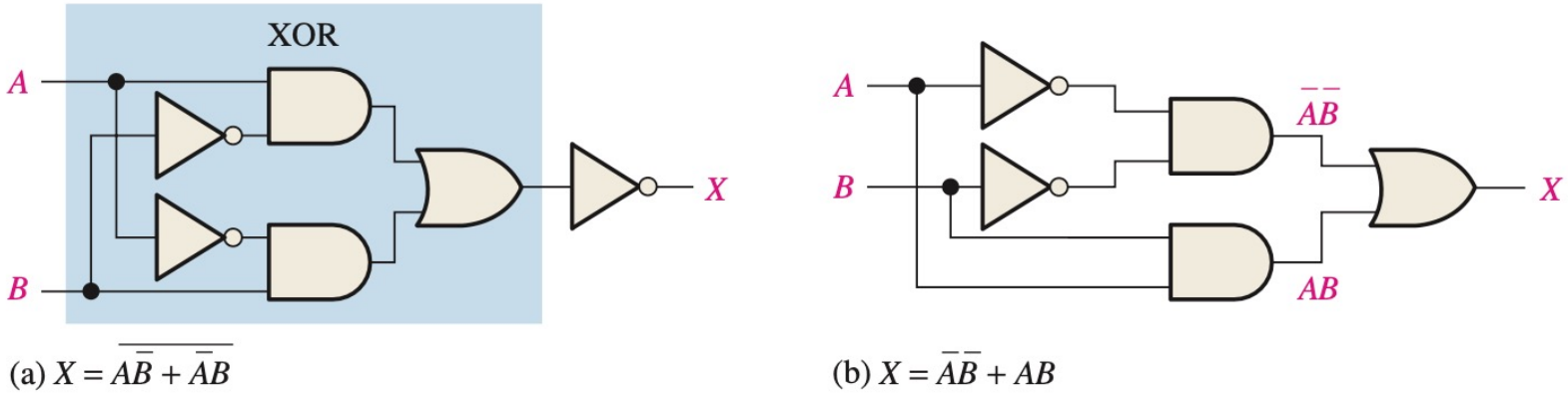
Truth table for an exclusive-OR.

| *A* | *B* | *X* |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Exclusive-NOR Logic

- As you know, 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) $X = A\overline{B} + \overline{A}B$ (b) $X = \overline{A}\,\overline{B} + AB$

**FIGURE 5–6** Two equivalent ways of implementing the exclusive-NOR. Open files F05-06 (a) and (b) to verify the operation.

MultiSim

# 2. Implementing Combinational Logic

## From a Boolean Expression to a Logic Circuit

- Let's examine the following Boolean expression:

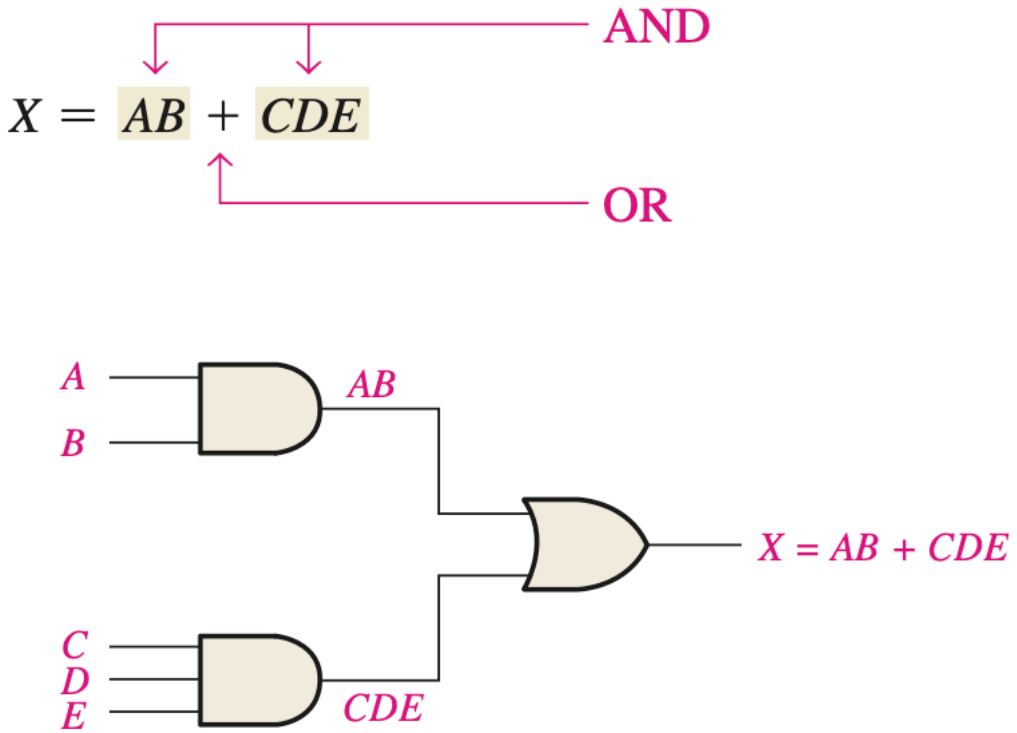
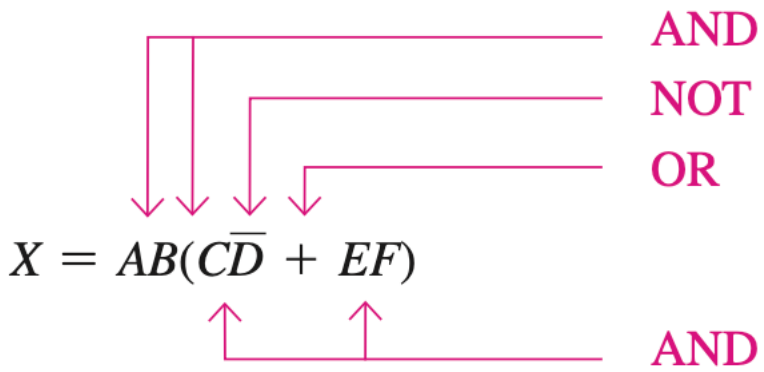



**FIGURE 5–9** Logic circuit for $X = AB + CDE$.

# From a Boolean Expression to a Logic Circuit

- Let's examine the following Boolean expression:




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

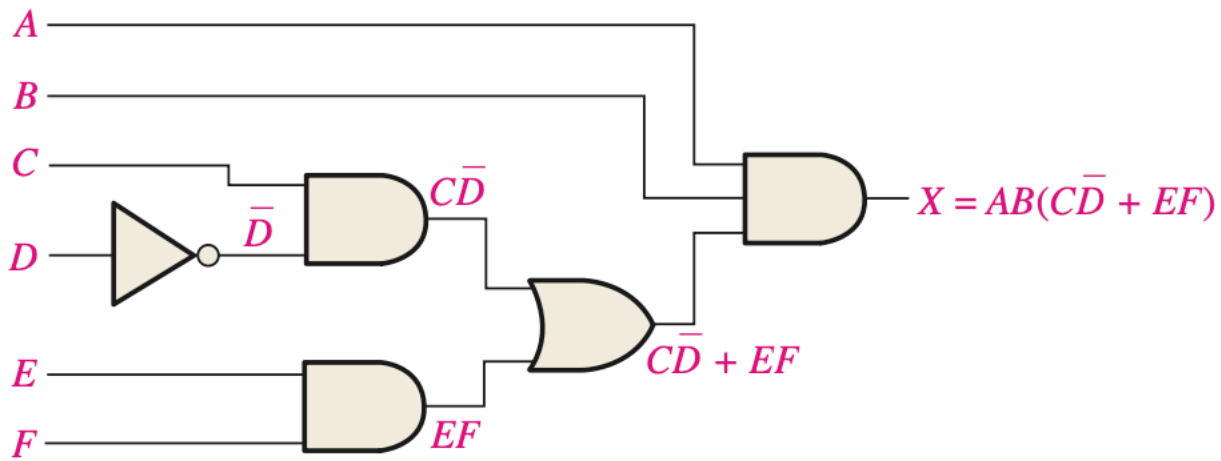
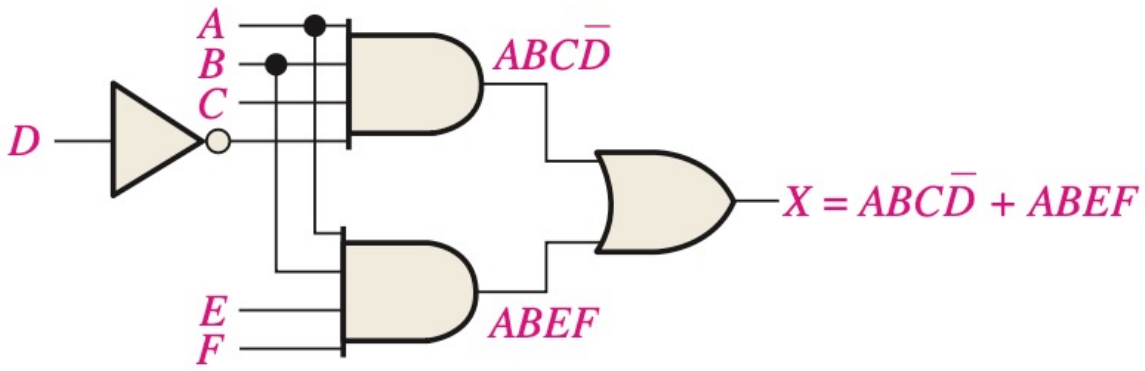



**FIGURE 5–10** Logic circuits for $X = AB(C\overline{D} + EF)$ :

# From a Boolean Expression to a Logic Circuit

- The expression is converted to SOP as follows:

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




(b) Sum-of-products implementation of the circuit in part (a)

**FIGURE 5–10** Logic circuits for $X = AB(C\overline{D} + EF) = ABC\overline{D} + ABEF$.

# From a Truth Table to a Logic Circuit

**TABLE 5–3**

| 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 | 1 | $A\overline{B}\,\overline{C}$ |
| 1 | 0 | 1 | 0 | |
| 1 | 1 | 0 | 0 | |
| 1 | 1 | 1 | 0 | |

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.

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

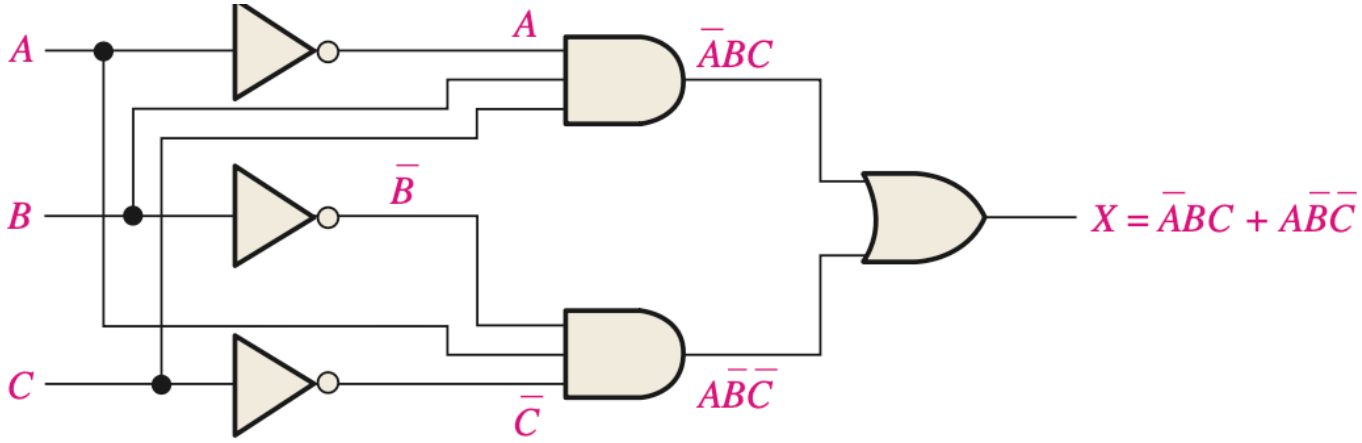
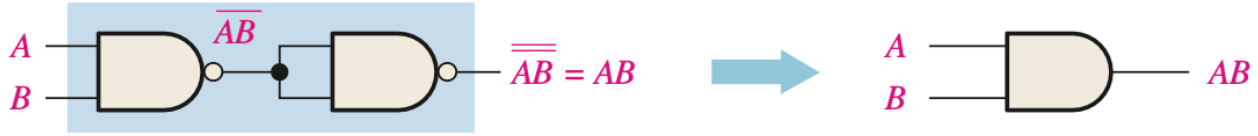



**FIGURE 5–11** Logic circuit for $X = \overline{A}BC + A\overline{B}\,\overline{C}$. Open file F05-11 to verify the operation.

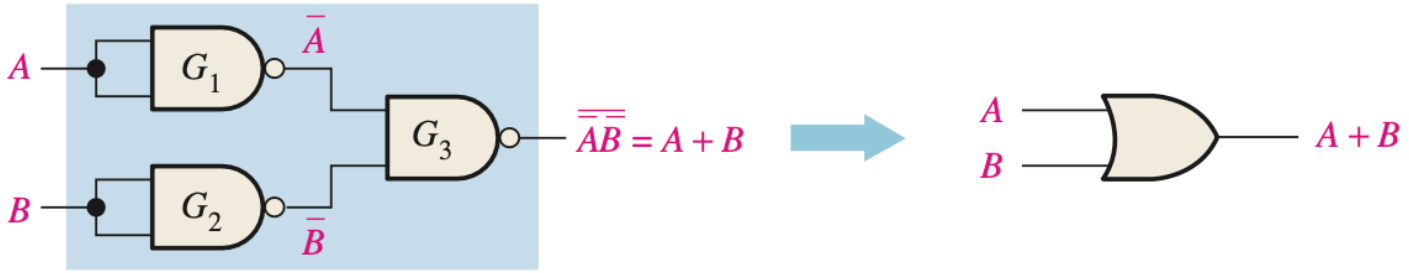
# 3. The Universal Property of NAND and NOR Gates

## The NAND Gate as a Universal Logic Element

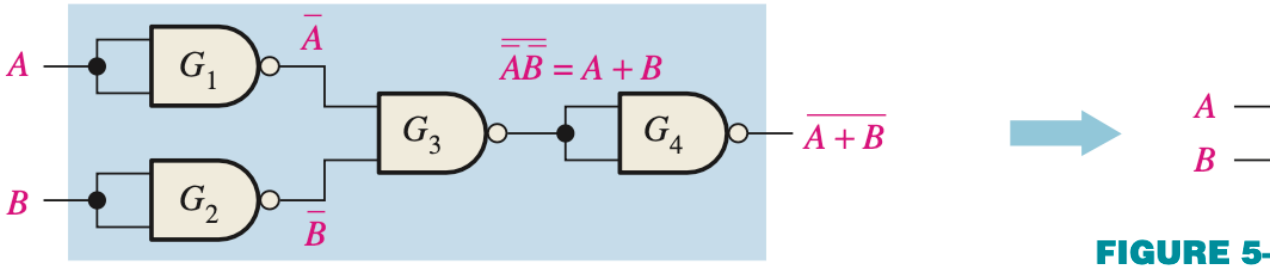

(a) One NAND gate used as an inverter




(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

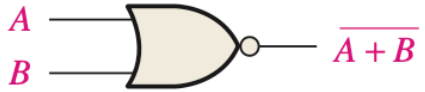
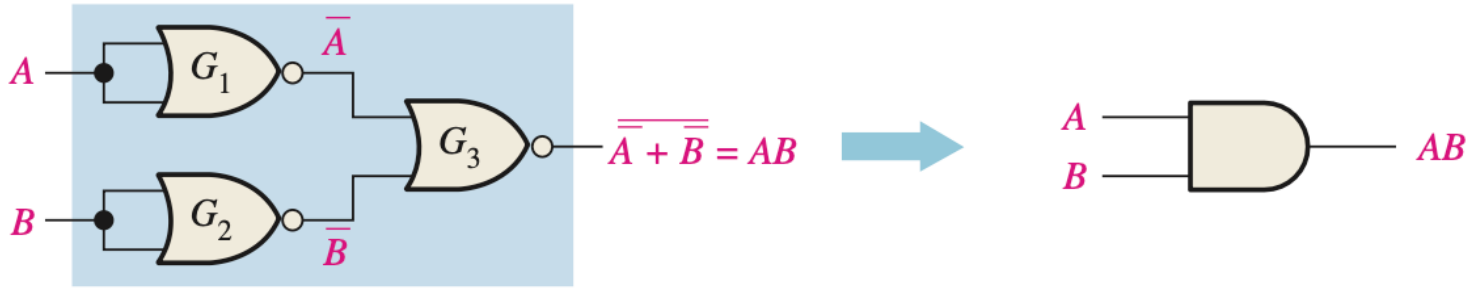



**FIGURE 5–18** Universal application of NAND gates. Open files F05-18(a), (b), (c), and (d) to verify each of the equivalencies.

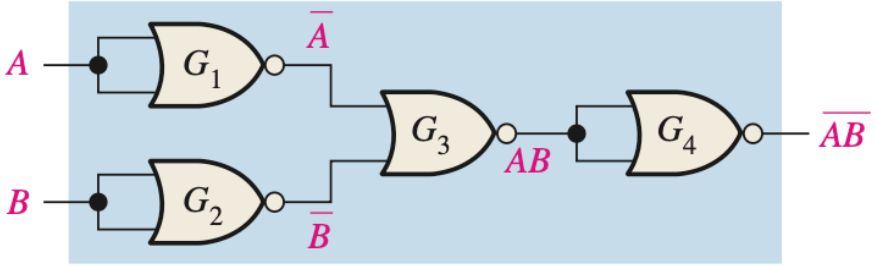
# The NOR Gate as a Universal Logic Element


(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

**FIGURE 5–19** Universal application of NOR gates. Open files F05-19(a), (b), (c), and (d) to verify each of the equivalencies.

# 4. Combinational Logic Using NAND and NOR Gates

## NAND Logic

- A NAND gate can function as either a NAND or a negative-OR because, by DeMorgan's theorem.

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

NAND — negative-OR

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

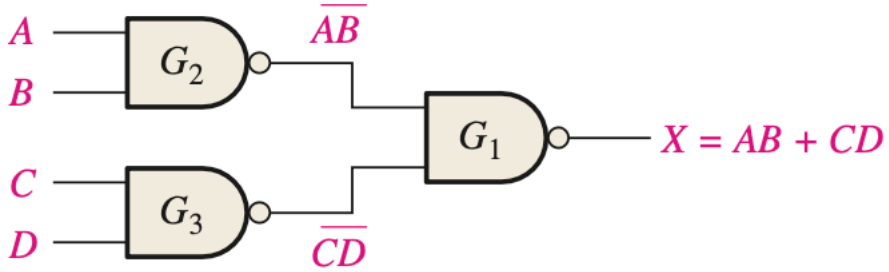
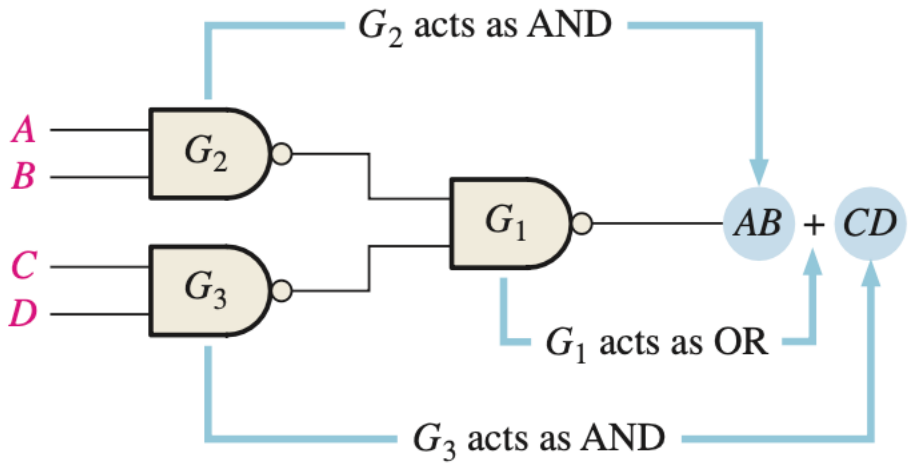
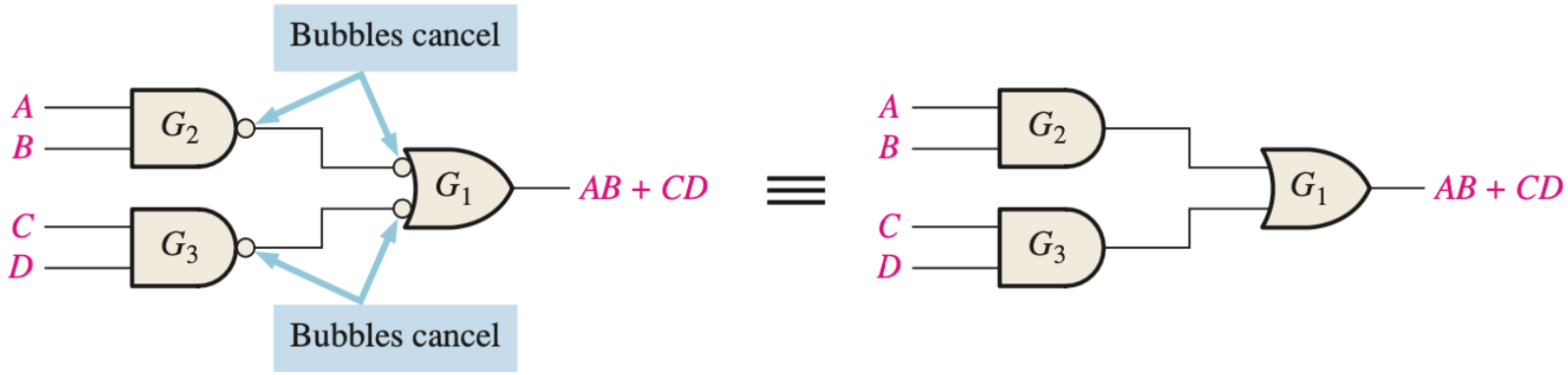



**FIGURE 5–20** NAND logic for $X = AB + CD$.

# NAND Logic




(a) Original NAND logic diagram showing effective

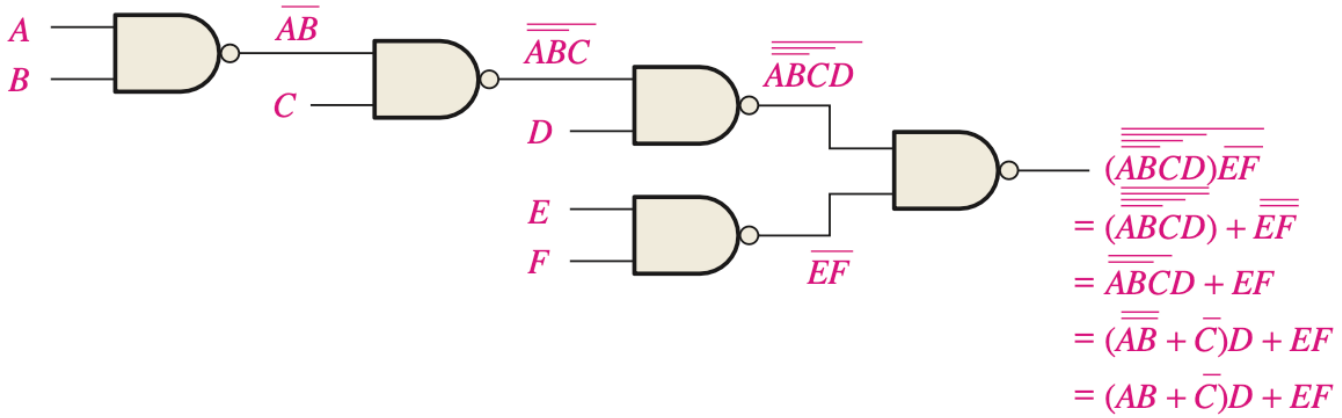



(b) Equivalent NAND/Negative-OR logic diagram (c) AND-OR equivalent

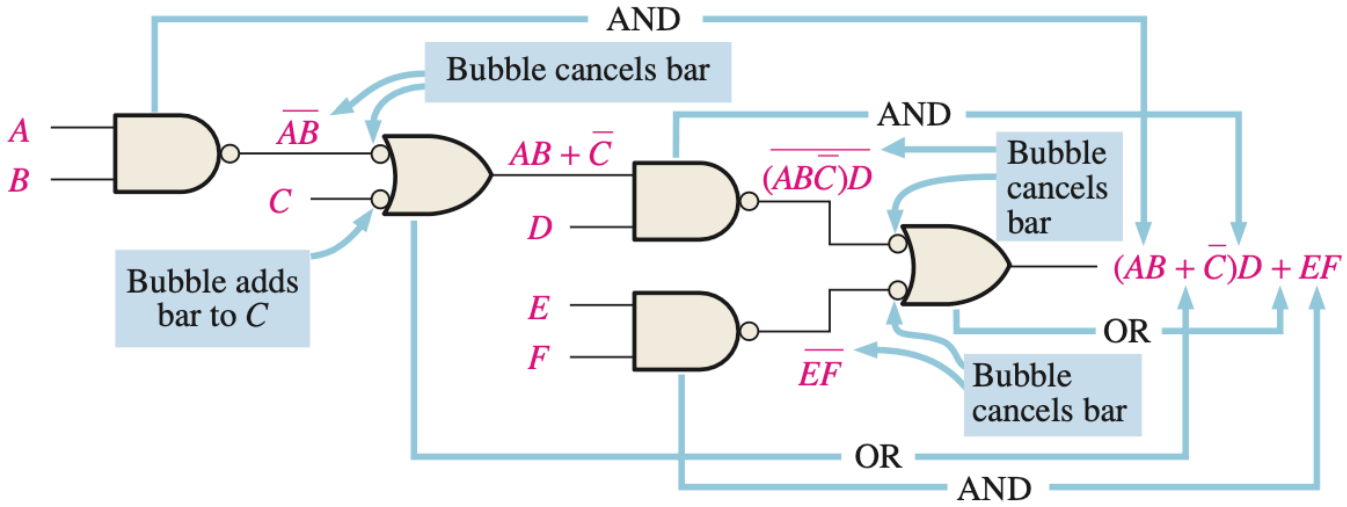
**FIGURE 5–21** Development of the AND-OR equivalent of the circuit in Figure 5–20.

Since a bubble represents an inversion, two connected bubbles represent a double inversion and therefore cancel each other.

# NAND Logic Diagrams Using Dual Symbols




(a) Several Boolean steps are required to arrive at final output expression.




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

**FIGURE 5–22** Illustration of the use of the appropriate dual symbols in a NAND logic diagram.

# NAND Logic Diagrams Using Dual Symbols

**EXAMPLE 5–9**

Redraw the logic diagram and develop the output expression for the circuit in Figure 5–23 using the appropriate dual symbols.

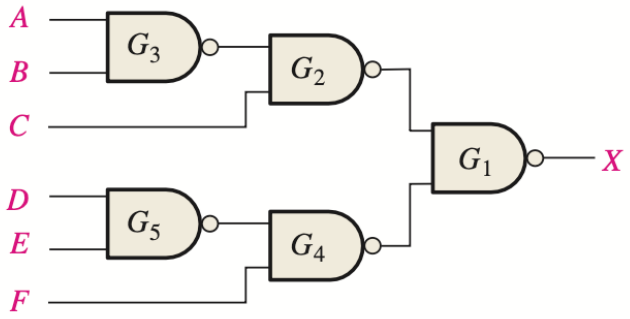



**FIGURE 5–23**

## Solution

Redraw the logic diagram in Figure 5–23 with the use of equivalent negative-OR symbols as shown in Figure 5–24. Writing the expression for $X$ directly from the indicated logic operation of each gate gives $X = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$.

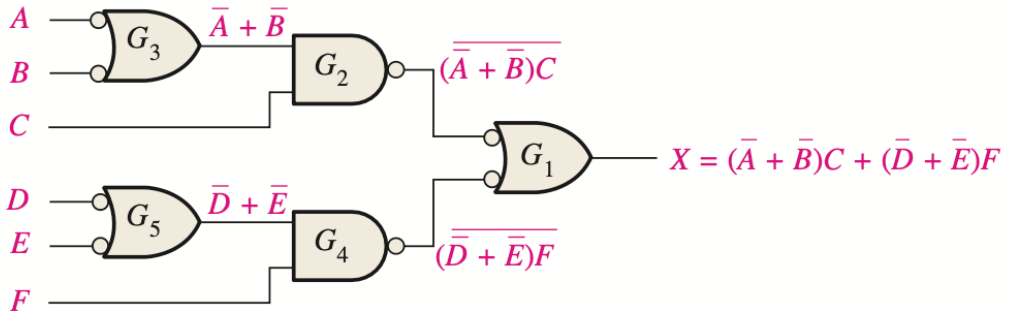



**FIGURE 5–24**

# NAND Logic Diagrams Using Dual Symbols

**EXAMPLE 5–10**

Implement each expression with NAND logic using appropriate dual symbols:

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

## Solution

See Figure 5–25.

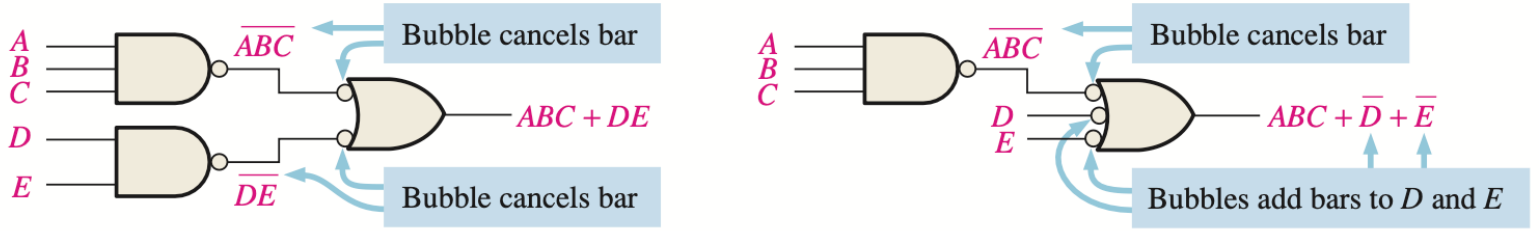



**FIGURE 5–25**

# NOR Logic

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

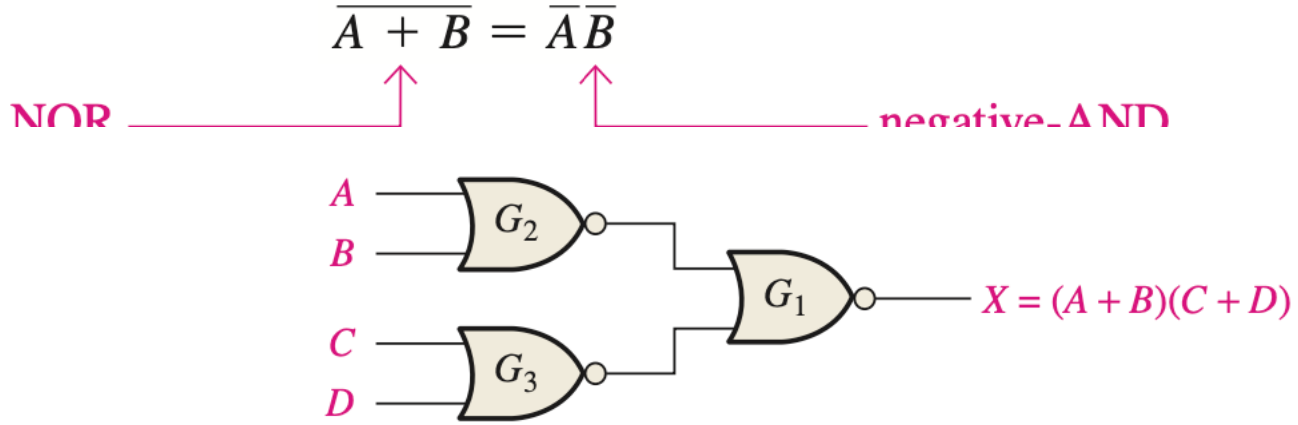



**FIGURE 5–26** NOR logic for $X = (A + B)(C + D)$.

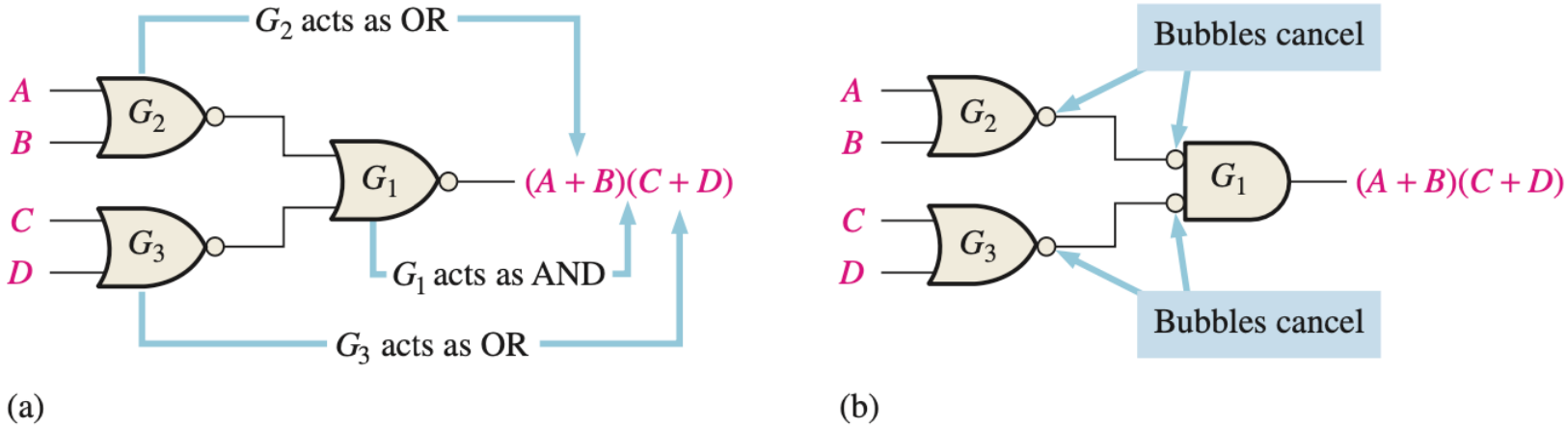
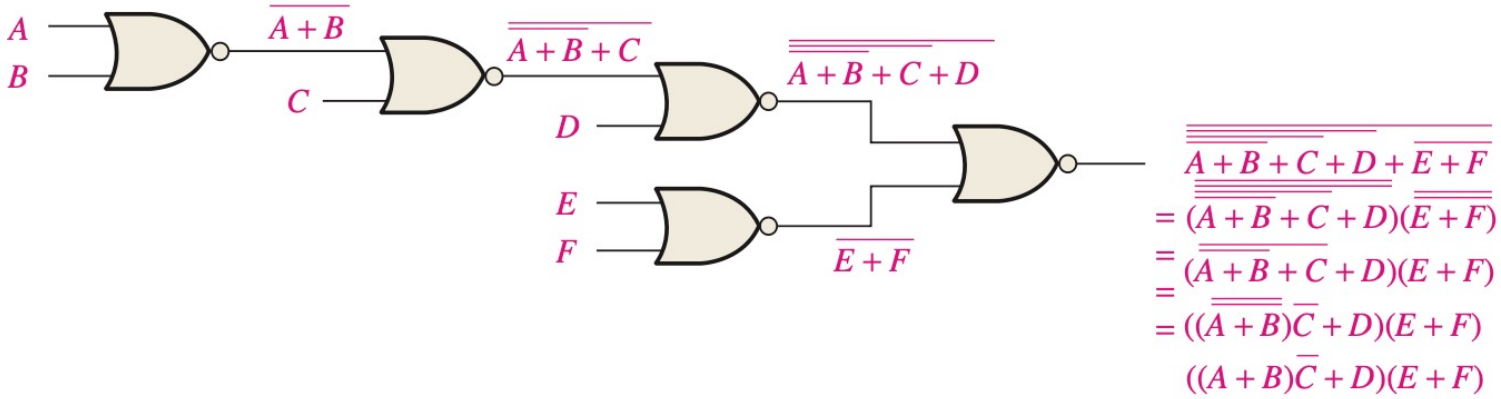
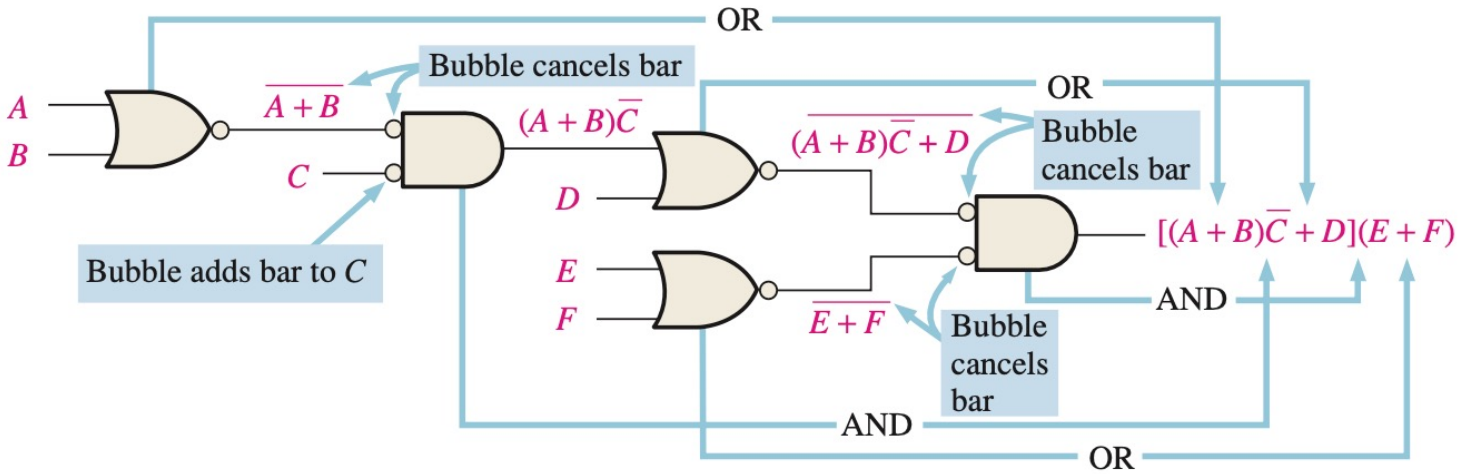



**FIGURE 5–27**

# NOR Logic Diagrams Using Dual Symbols




(a) Final output expression is obtained after several Boolean steps.




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

**FIGURE 5–28** Illustration of the use of the appropriate dual symbols in a NOR logic diagram.

# NOR Logic Diagrams Using Dual Symbols

**EXAMPLE 5–11**

Using appropriate dual symbols, redraw the logic diagram and develop the output expression for the circuit in Figure 5–29.

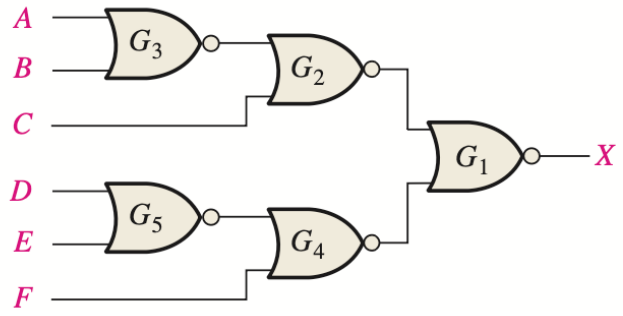



**FIGURE 5–29**

## Solution

Redraw the logic diagram with the equivalent negative-AND symbols as shown in Figure 5–30. Writing the expression for $X$ directly from the indicated operation of each gate,

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

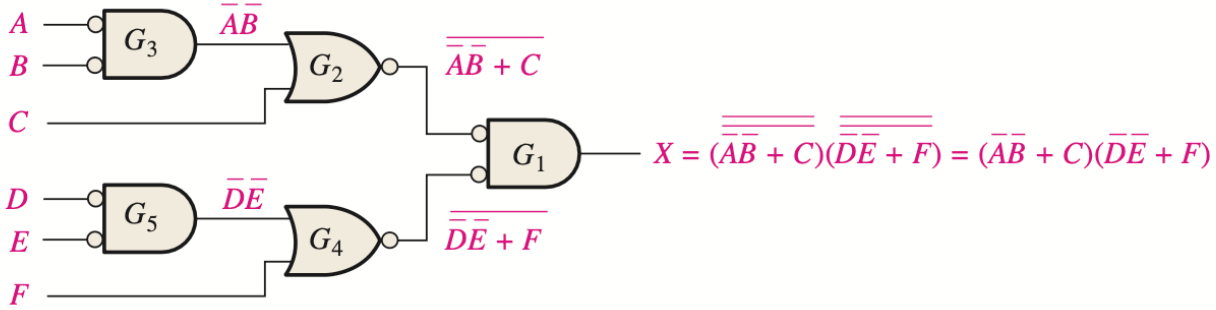



**FIGURE 5–30**

# 5. Pulse Waveform Operation

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

- The output of an AND gate is HIGH only when all inputs are HIGH at the same time.
- The output of an OR gate is HIGH only when at least one of its inputs is HIGH.
- The output of a NAND gate is LOW only when all inputs are HIGH at the same time.
- The output of a NOR gate is LOW only when at least one of its inputs is HIGH.

## EXAMPLE 5–12

Determine the final output waveform $X$ for the circuit in Figure 5–31, with input waveforms $A$, $B$, and $C$ as shown.

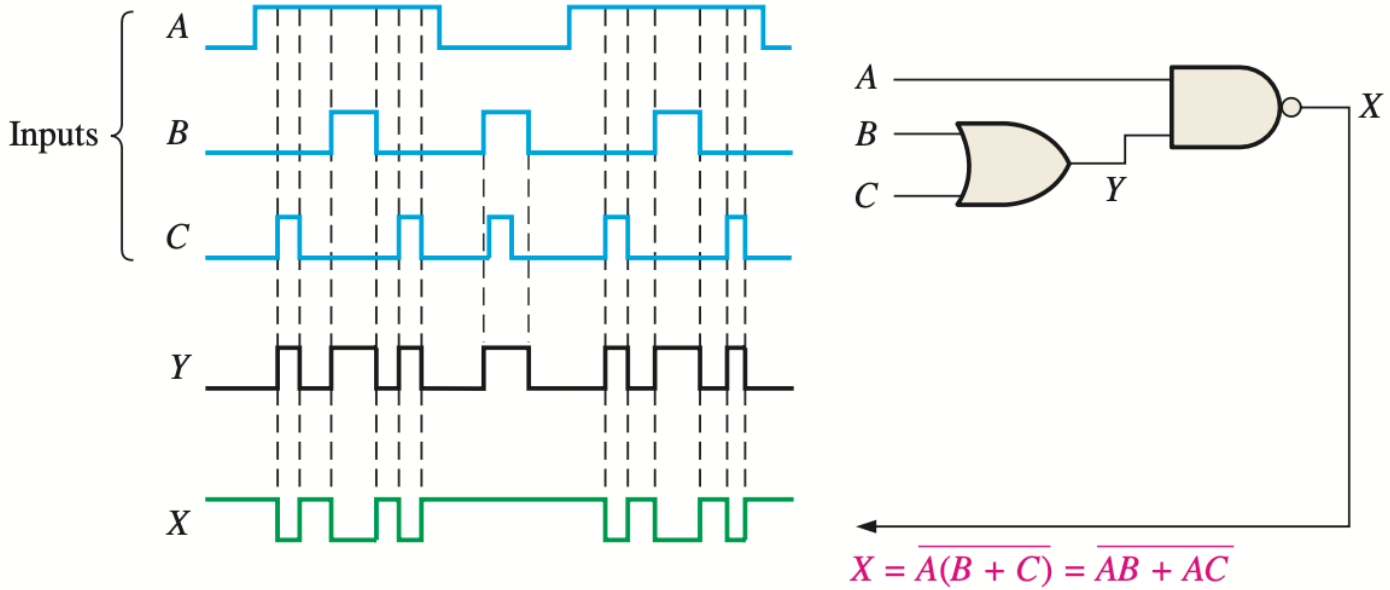



**FIGURE 5–31**

## Solution

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. The output waveform $X$ is shown in the timing diagram of Figure 5–31. The intermediate waveform $Y$ at the output of the OR gate is also shown.

## EXAMPLE 5–13

Draw the timing diagram for the circuit in Figure 5–32 showing the outputs of $G_1$, $G_2$, and $G_3$ with the input waveforms, $A$, and $B$, as indicated.

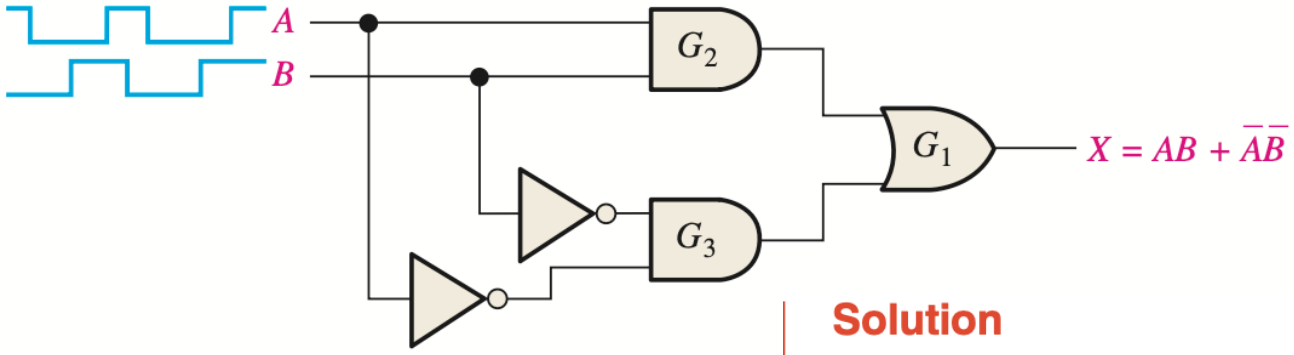



**FIGURE 5–32**

### Solution

When both inputs are HIGH or when both inputs are LOW, the output $X$ is HIGH as shown in Figure 5–33. Notice that this is an exclusive-NOR circuit. The intermediate outputs of gates $G_2$ and $G_3$ are also shown in Figure 5–33.

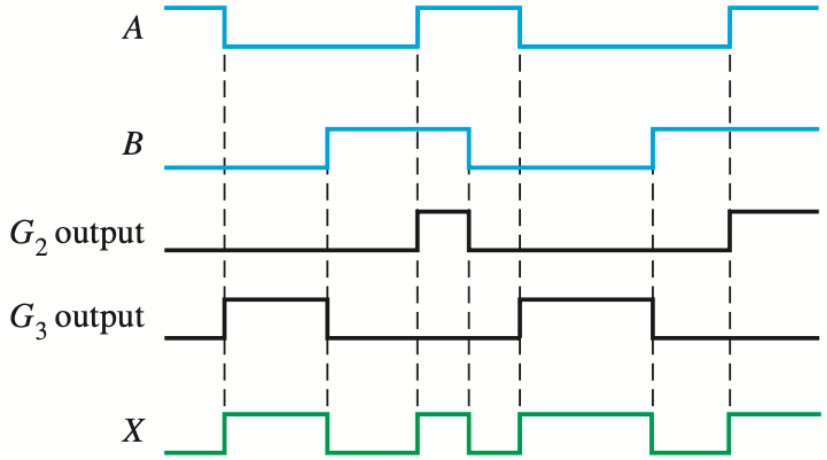
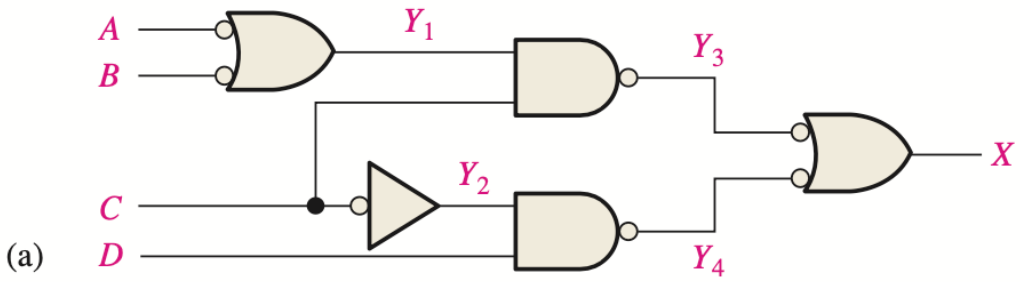


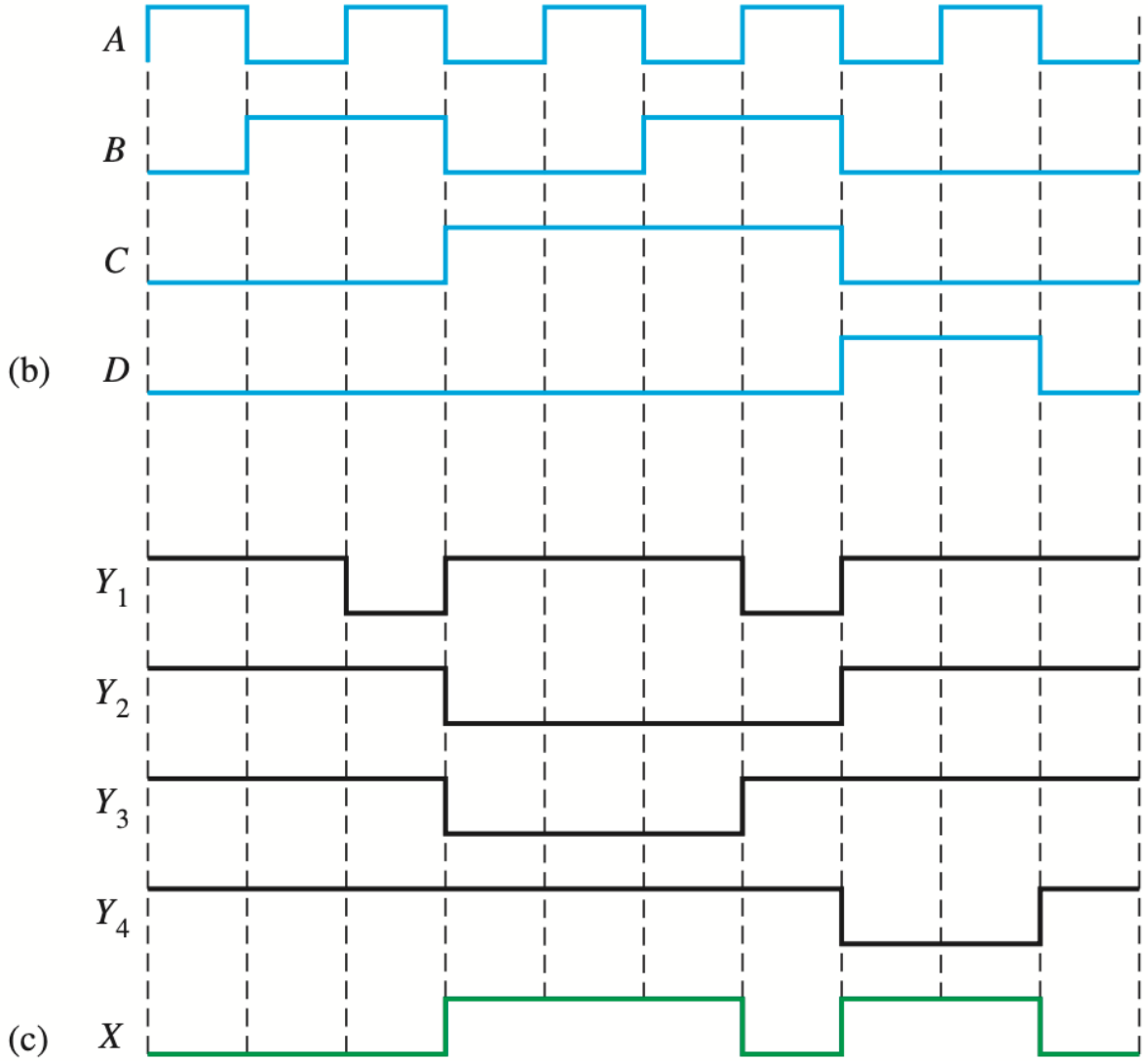

**FIGURE 5–33**

## EXAMPLE 5–14

Determine the output waveform $X$ for the logic circuit in Figure 5–34(a) by first finding the intermediate waveform at each of points $Y_1$, $Y_2$, $Y_3$, and $Y_4$. The input waveforms are shown in Figure 5–34(b).




(a)




(b)

(c)

**FIGURE 5–34**

**EXAMPLE 5–15**

Determine the output waveform $X$ for the circuit in Example 5–14, Figure 5–34(a), directly from the output expression.

## Solution

The output expression for the circuit is developed in Figure 5–35. The SOP form indicates that the output is HIGH when $A$ is LOW and $C$ is HIGH or when $B$ is LOW and $C$ is HIGH or when $C$ is LOW and $D$ is HIGH.

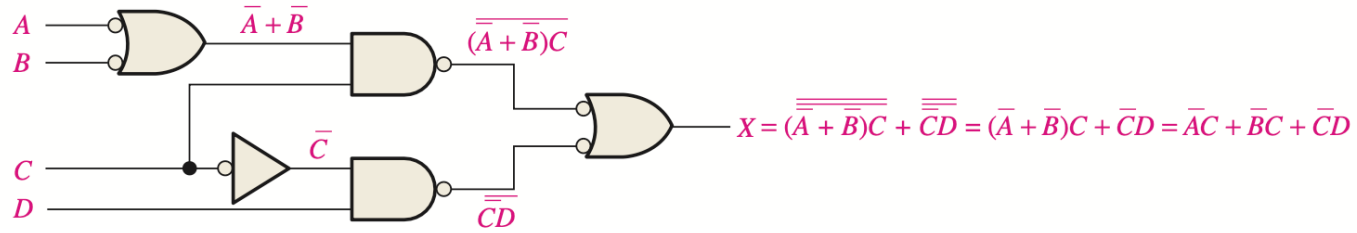



**FIGURE 5–35**

The result is shown in Figure 5–36 and is the same as the one obtained by the intermediate-waveform method in Example 5–14. The corresponding product terms for each waveform condition that results in a HIGH output are indicated.

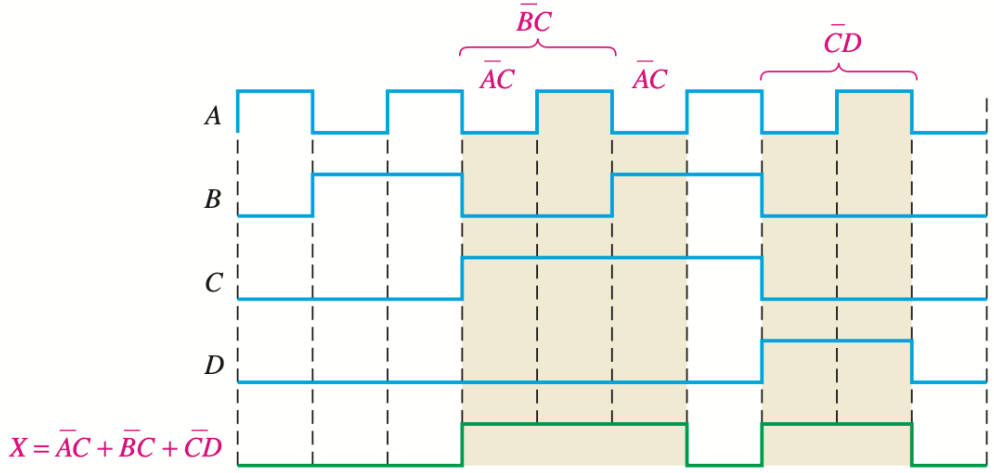
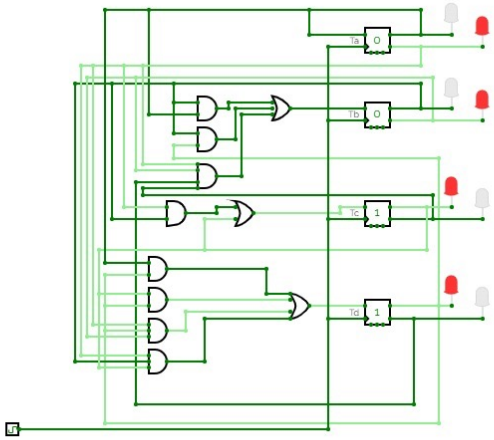



**FIGURE 5–36**

# THE END

## Lecture 5: Combinational Logic Analysis



INSTRUCTOR: Dr. Vuong Quoc Bao