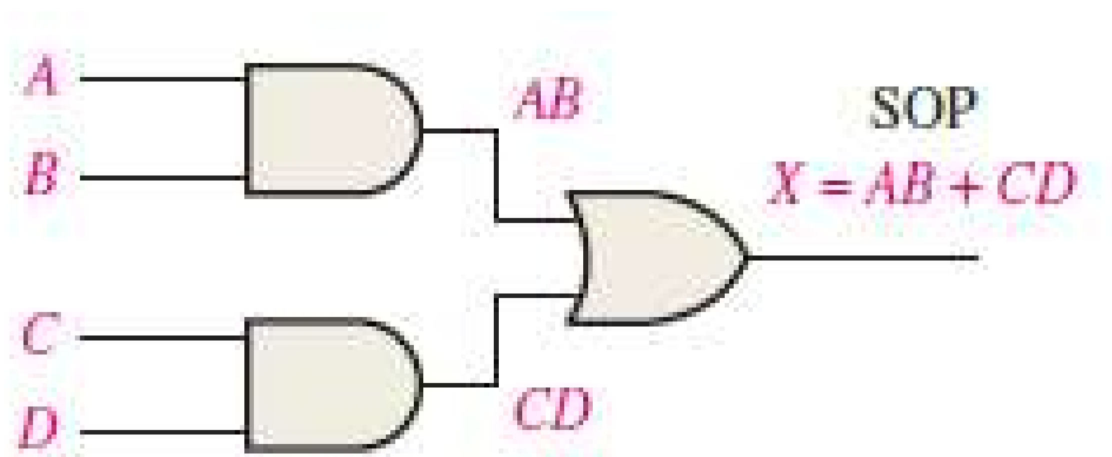
Combinational Circuits

AND-OR Logic

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



AND-OR Logic

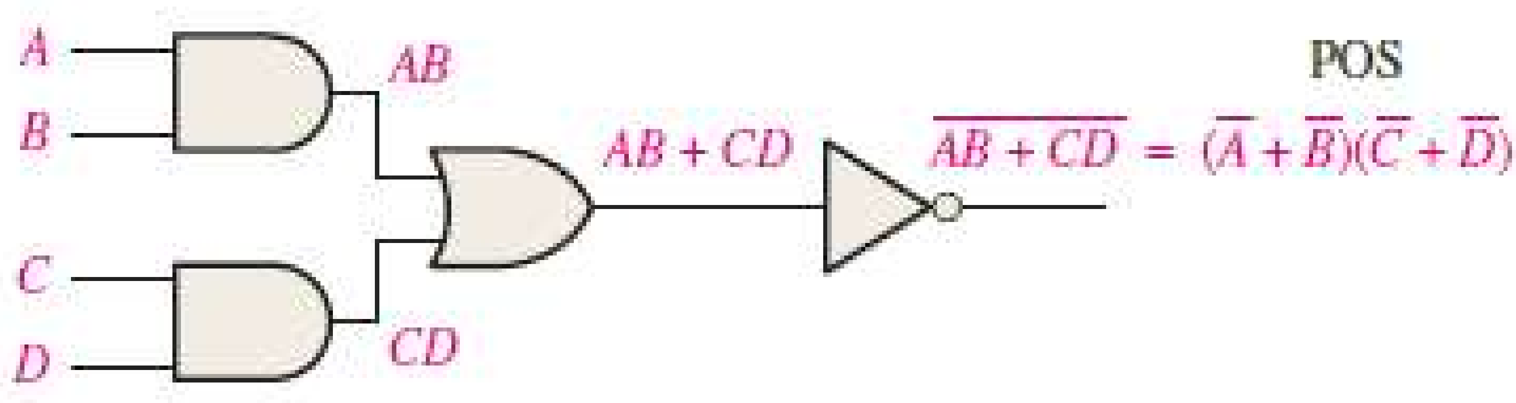
- The truth table for a 4-input AND-OR logic circuit is shown in the Table below.
- The intermediate AND gate outputs (the *AB and CD columns) are also shown in the table.*

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

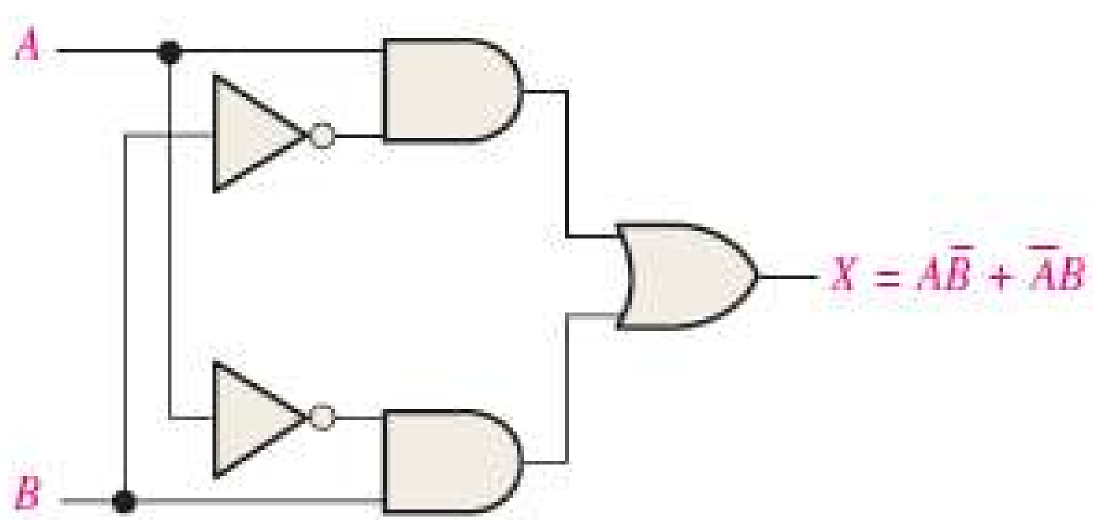
- When the output of an AND-OR circuit is complemented (inverted), it results in an AND-OR-Invert circuit.

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



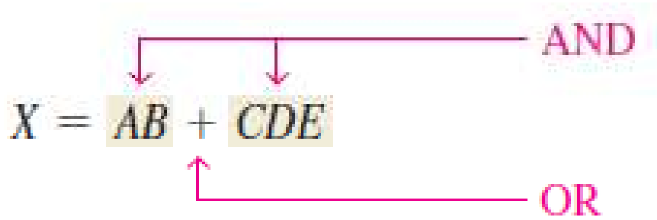
Exclusive-OR Logic

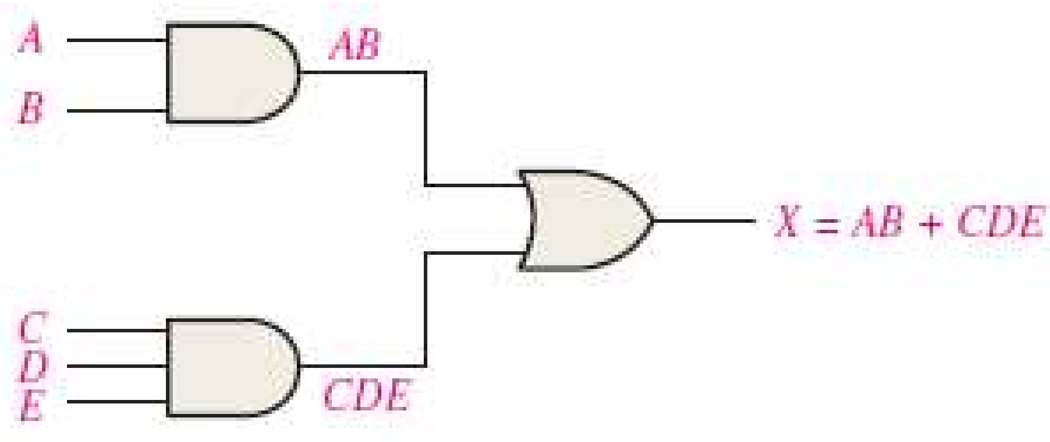
- Although this circuit is considered a type of logic gate with its own unique symbol, it is actually a combination of two AND gates, one OR gate, and two inverters, as shown below



From a Boolean Expression to a Logic Circuit

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



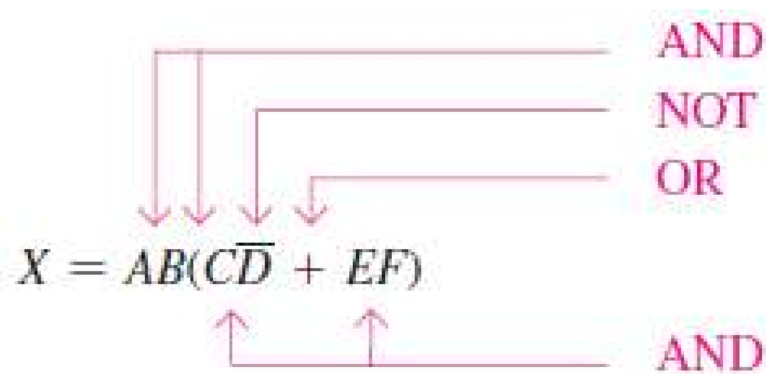


From a Boolean Expression to a Logic Circuit

- let’s implement the following expression:

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

- A breakdown of this expression shows that the terms *AB and* $\left(C\overline{D} + EF\right)$ are ANDed.
- The term $\left(C\overline{D} + EF\right)$ is formed by first ANDing *C and* $\overline{D}$ and ANDing *E* and *F,* and the Oring these two terms.



From a Boolean Expression to a Logic Circuit

- Before you can implement the final expression, you must create the sum term

$(C\overline{D} + EF)$ but before you can get this term; you must create the product terms $C\overline{D}$ and *EF;*

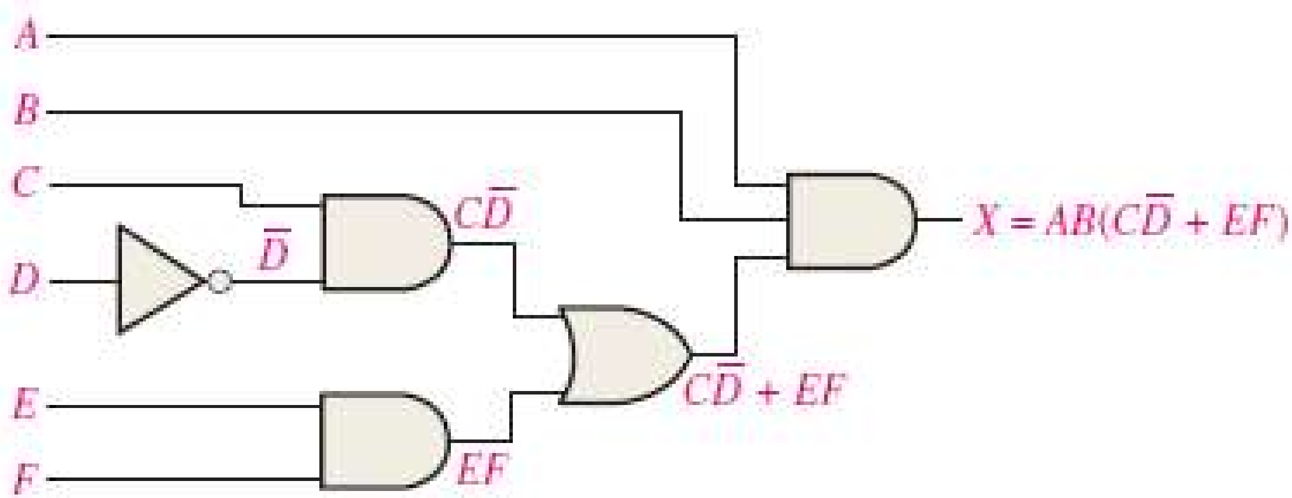
- but before you can get the term $C\overline{D}$ *you must create* $\overline{D}$*.*
- The logic gates required to implement

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

are as follows

From a Boolean Expression to a Logic Circuit

- One inverter to form $\overline{D}$
- Two 2-input AND gates to form $C\overline{D}$ and *EF*
- One 2-input OR gate to form $(C\overline{D} + EF)$
- One 3-input AND gate to form *X*



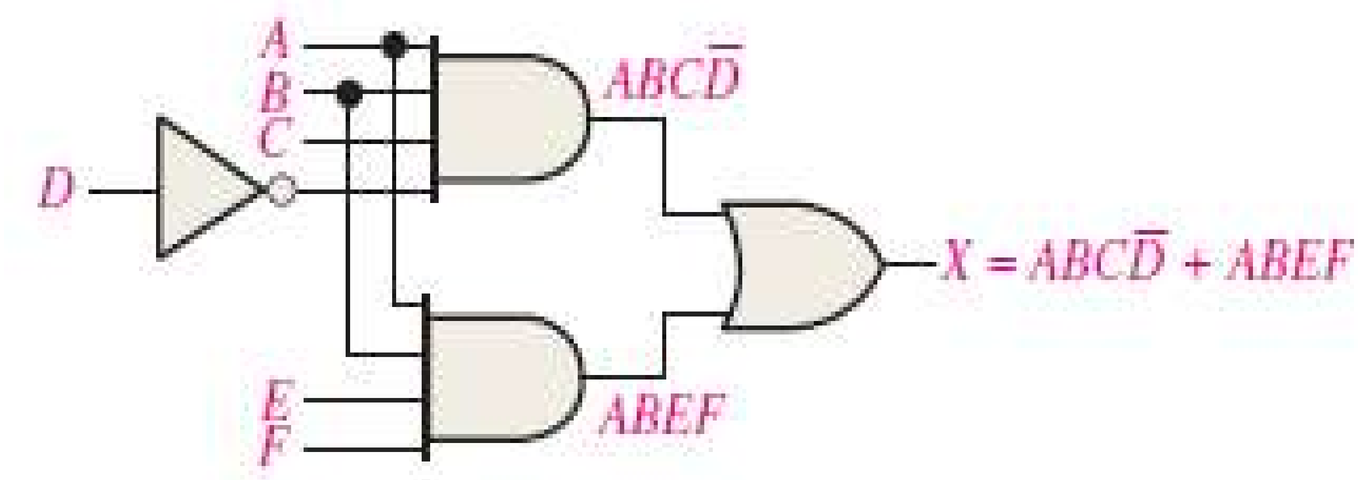
From a Boolean Expression to a Logic Circuit

- Notice that there is a maximum of four gates and an inverter between an input and output in this circuit (from input D to output).
- Often the total propagation delay time through a logic circuit is a major consideration.
- Propagation delays are additive, so the more gates or inverters between input and output, the greater the propagation delay time.

From a Boolean Expression to a Logic Circuit

- It is usually best to reduce a circuit to its SOP form in order to reduce the overall propagation delay time.

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



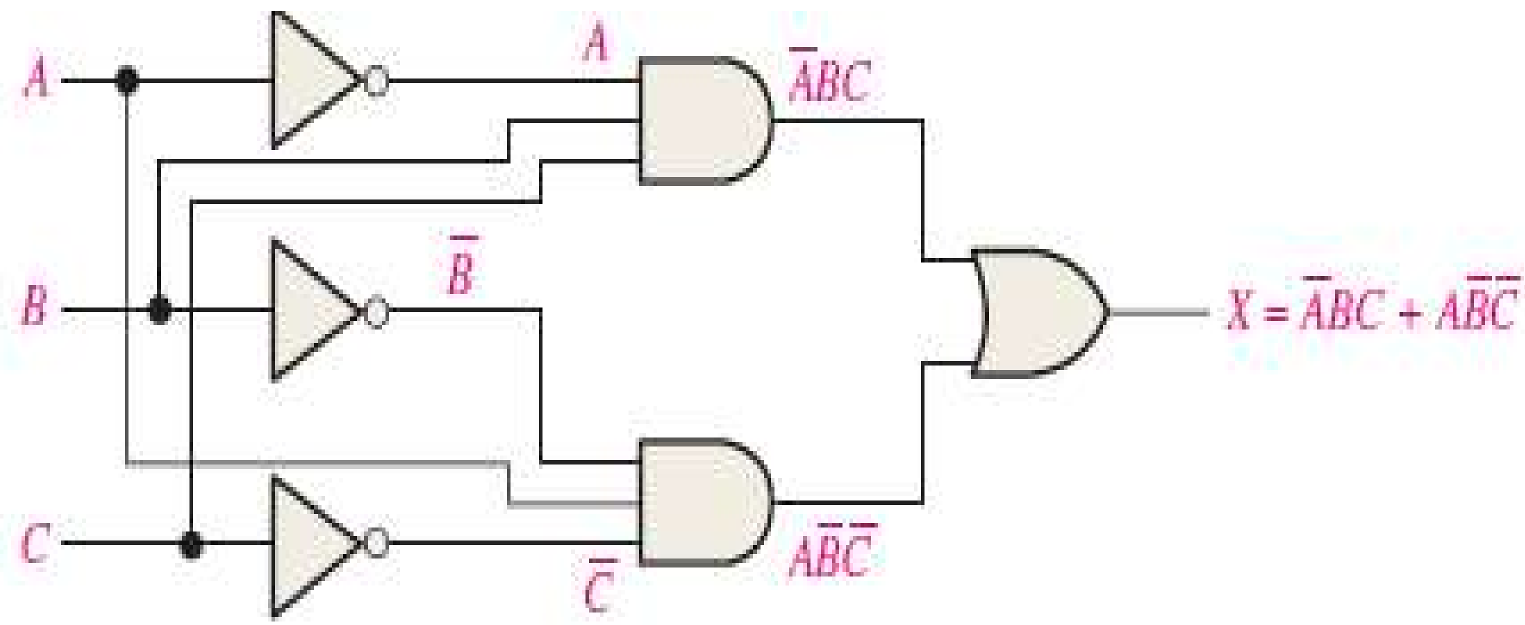
From a Truth Table to a Logic Circuit

- 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.

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	

From a Truth Table to a Logic Circuit

- The Boolean SOP expression obtained from the truth table by ORing the product terms
- for which *X = 1 is* $X = \overline{A}BC + A\overline{B}\overline{C}$



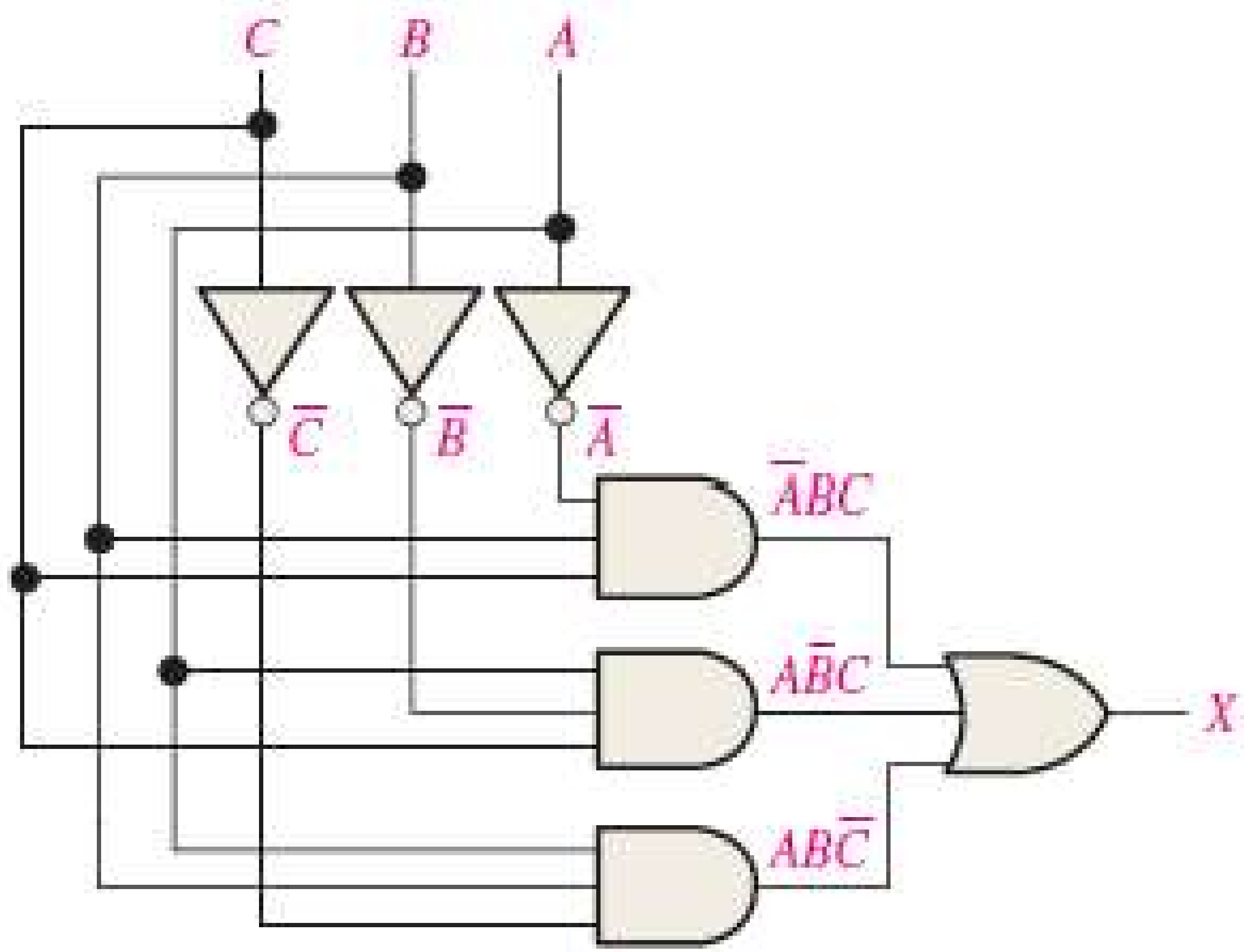
From a Truth Table to a Logic Circuit

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

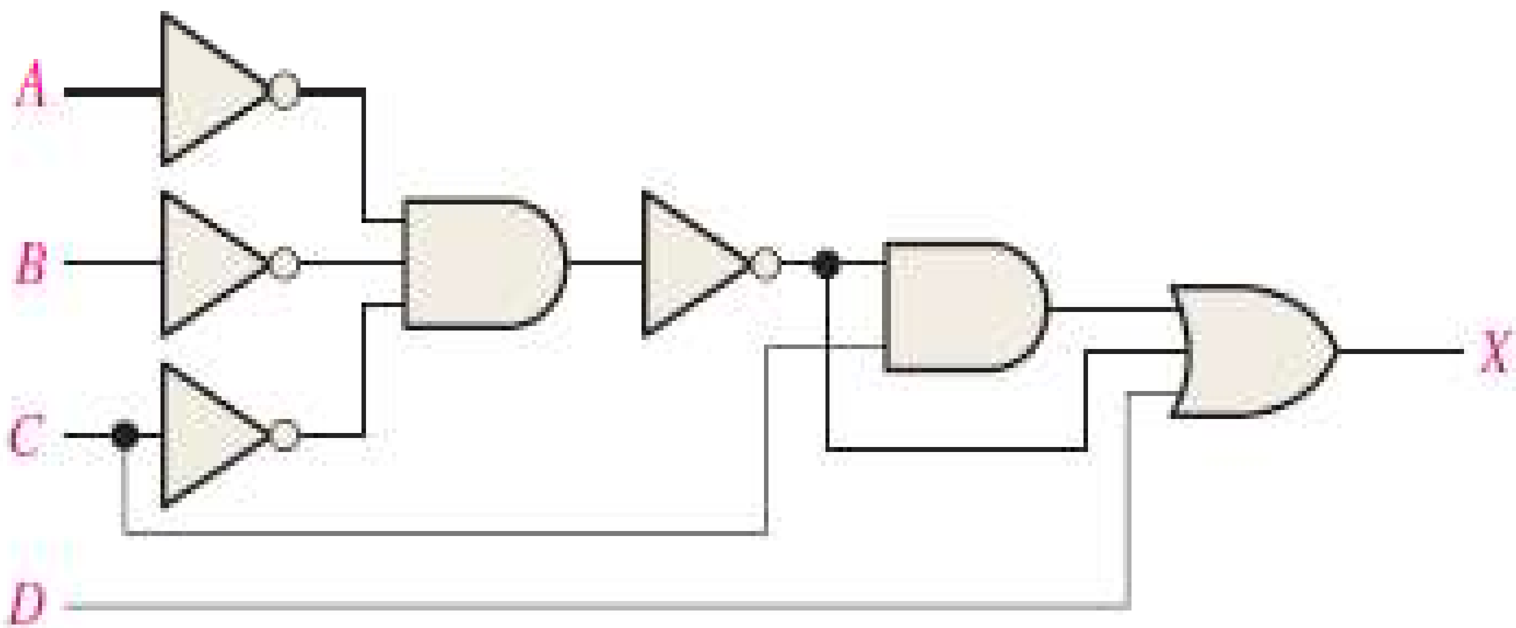
Inputs			Output	
A	B	C	X	Product Term
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	0	

From a Truth Table to a Logic Circuit

- Notice that *X = 1 for only three of the input conditions. Therefore, the logic expression is*
  $X = \overline{A}BC + A\overline{B}C + AB\overline{C}$
- The logic gates required are three inverters, three 3-input AND gates and one 3-input OR gate.


C
B
A
$\overline{C}$
$\overline{B}$
$\overline{A}$
$\overline{A}BC$
$A\overline{B}C$
$AB\overline{C}$
X

- Reduce the combinational logic circuit in 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,

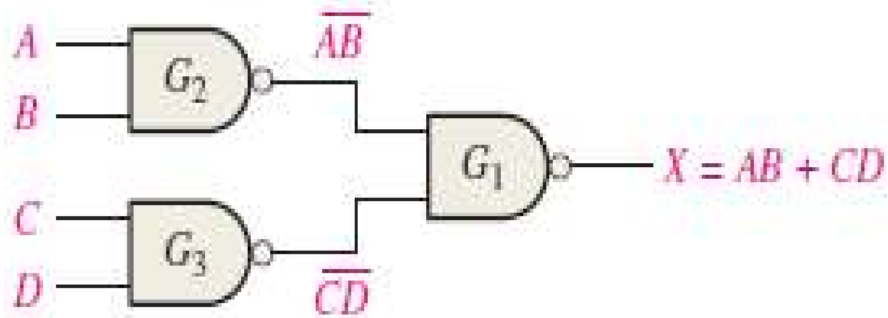
$$\begin{aligned}
X &= (\overline{\overline{A}} + \overline{\overline{B}} + \overline{\overline{C}})C + \overline{\overline{A}} + \overline{\overline{B}} + \overline{\overline{C}} + D \\
&= AC + BC + CC + A + B + C + D \\
&= AC + BC + C + A + B + \cancel{C} + D \\
&= C(A + B + 1) + A + B + D \\
X &=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

NAND Logic

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

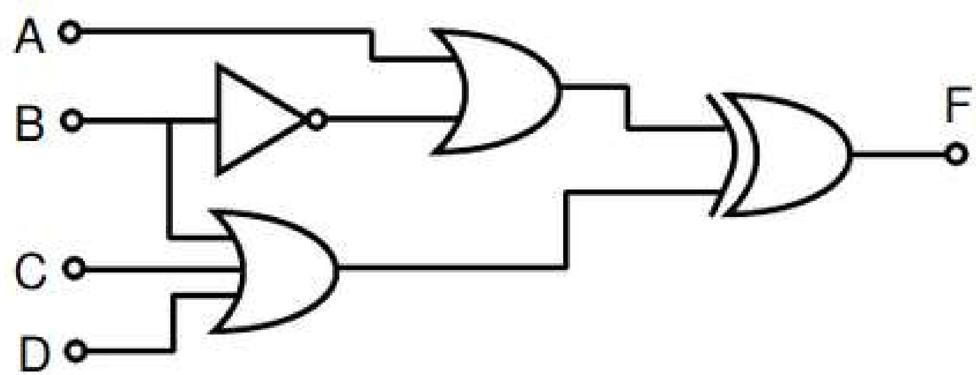
NAND — negative-OR

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



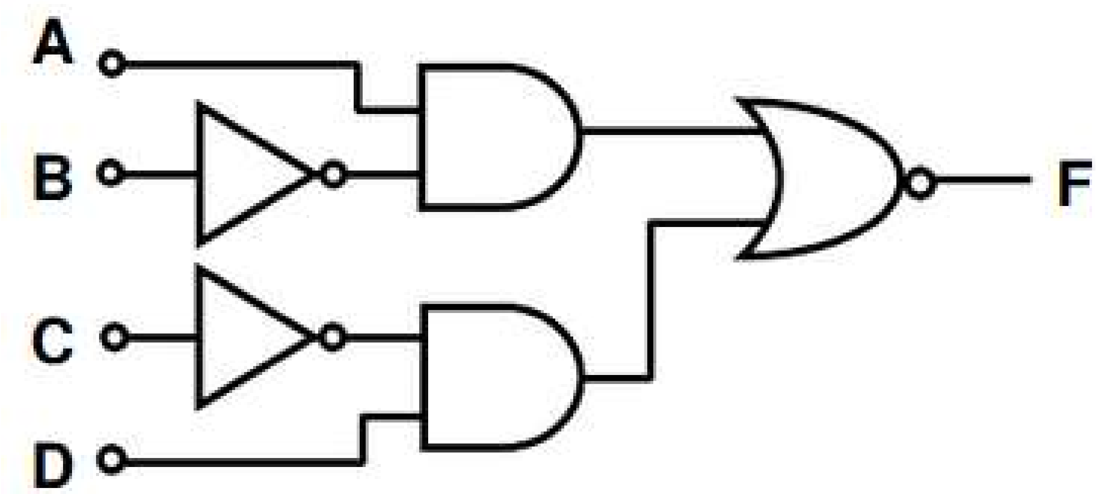
Try 1

- Determine the output (in terms of Boolean expressions) F.
- Complete the truth table for the function F.
- From the truth table, write the ‘sum of products’ expression for the function F



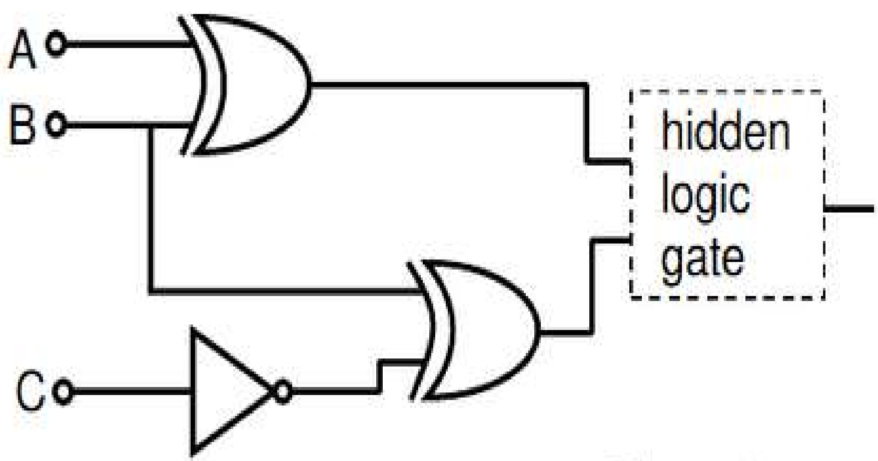
Try 2

- Determine the output F in the circuit shown
- By utilizing Boolean algebra, show that the function F obtained above is also equivalent to $\left(\overline{A}+B\right)\bullet\left(C+\overline{D}\right)$



Try 3

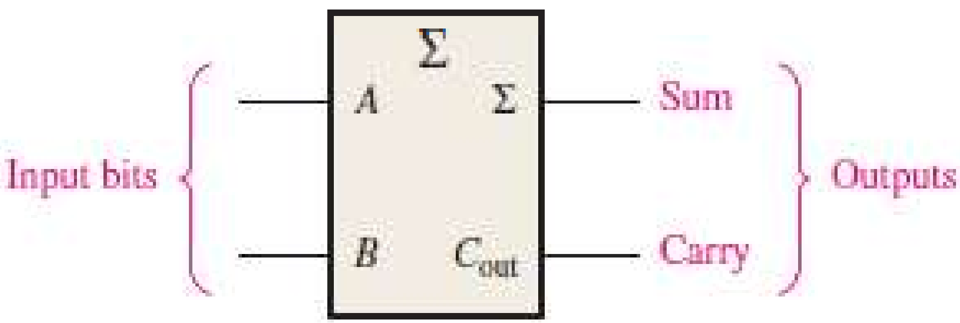
- The circuit shown below in Figure 3 has a two-input logic gate hidden from view. By inspection of the output function F, identify the hidden logic gate
- Draw a truth table for the function F given in part (A) above.
- Derive an alternative 'sum of products' expression for F



$$F = \overline{(A \oplus B) + (\bar{C} \oplus B)}$$

The Half-Adder

- The half-adder accepts two binary digits on its inputs and produces two binary digits on its outputs—a sum bit and a carry bit.



The Half-Adder

Half-adder truth table.

A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Σ = sum
C_{out} = output carry
A and B = input variables (operands)

The Half-Adder

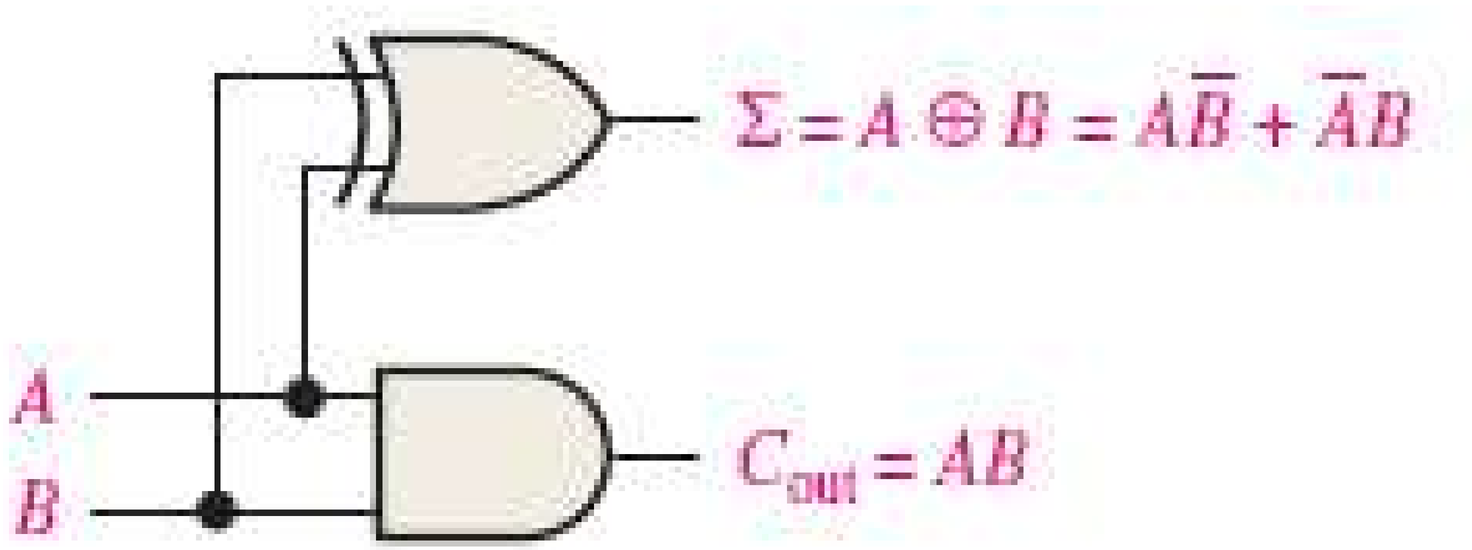
- Notice that the output carry (*Cout)* is a 1 only when both *A and B are 1s; therefore, Cout can be expressed as the AND of the* input variables.

$$C_{out} = AB$$

- Now observe that the sum output is a 1 only if the input variables, *A and B, are not* equal. The sum can therefore be expressed as the exclusive-OR of the input variables.

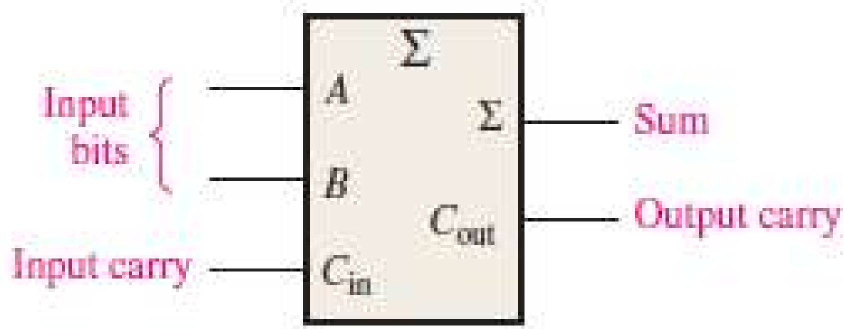
$$\sum = A \oplus B$$

The Half-Adder



The Full-Adder

- The full-adder accepts two input bits and an input carry and generates a sum output and an output carry.



The Full-Adder

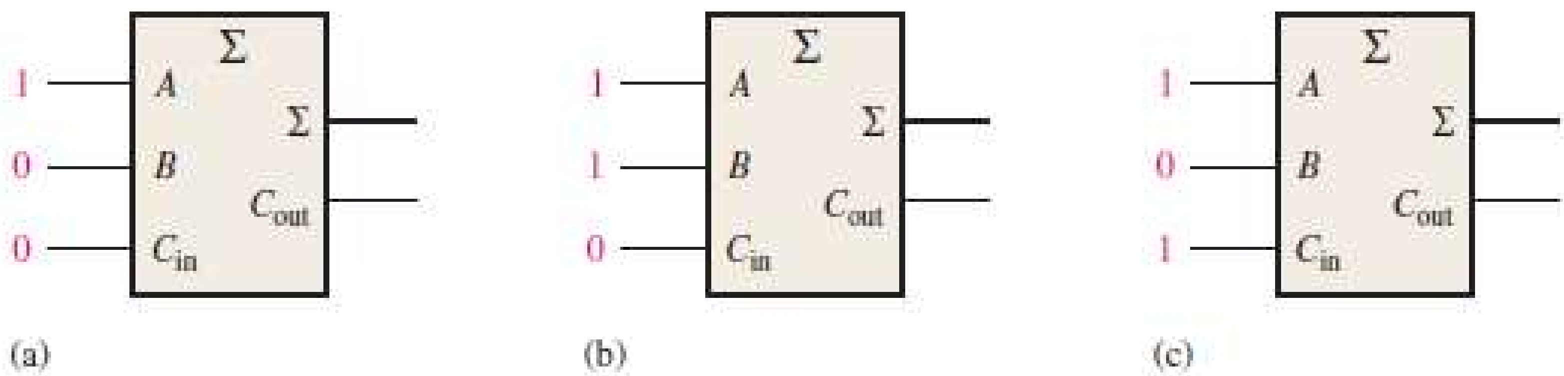
Full-adder truth table.

A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

C_{in} = input carry, sometimes designated as CI
C_{out} = output carry, sometimes designated as CO
Σ = sum
A and B = input variables (operands)

The Full-Adder Logic

- For each of the three full-adders determine the outputs for the inputs shown.



The Full-Adder Logic

(a) The input bits are $A = 1$, $B = 0$, and $C_{in} = 0$.

$$1 + 0 + 0 = 1 \text{ with no carry}$$

Therefore, $\Sigma = \mathbf{1}$ and $C_{out} = \mathbf{0}$.

(b) The input bits are $A = 1$, $B = 1$, and $C_{in} = 0$.

$$1 + 1 + 0 = 0 \text{ with a carry of } 1$$

Therefore, $\Sigma = \mathbf{0}$ and $C_{out} = \mathbf{1}$.

(c) The input bits are $A = 1$, $B = 0$, and $C_{in} = 1$.

$$1 + 0 + 1 = 0 \text{ with a carry of } 1$$

Therefore, $\Sigma = \mathbf{0}$ and $C_{out} = \mathbf{1}$.