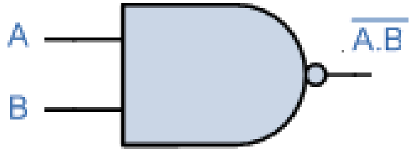
NAND AND NOR IMPLEMENTATION

Universal Gates

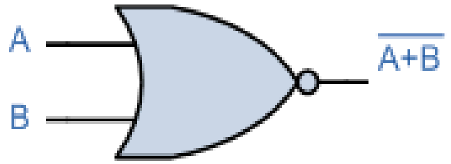
- A universal gate is a gate that can implement any Boolean function without the need to use any other gate type.
- The NAND and NOR gates are universal gates.
- The NAND and NOR gates are said to be *universal* gates because any logic circuit can be implemented with it.

NAND Gate



INPUT		OUTPUT
A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

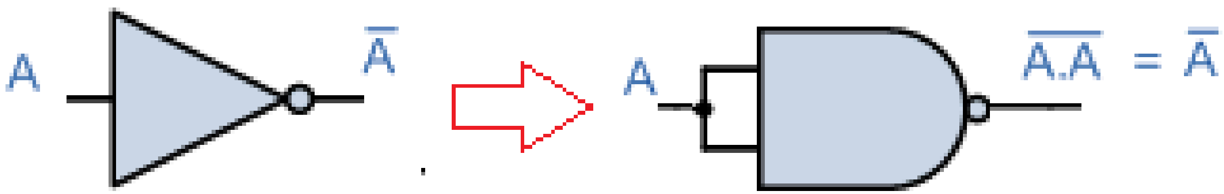
NOR Gate



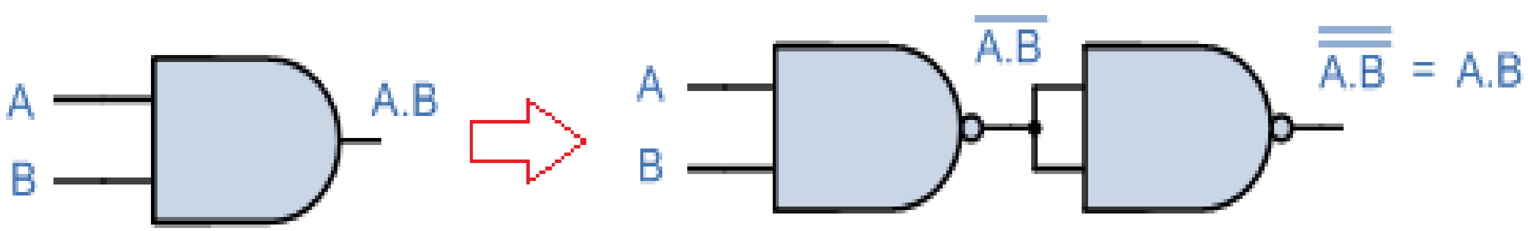
INPUT		OUTPUT
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Logic Gates using only NAND Gates

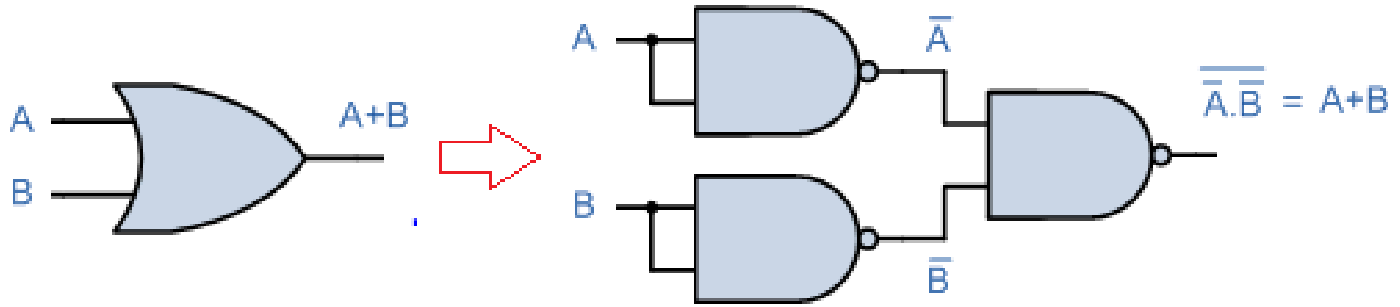
- NOT Gate



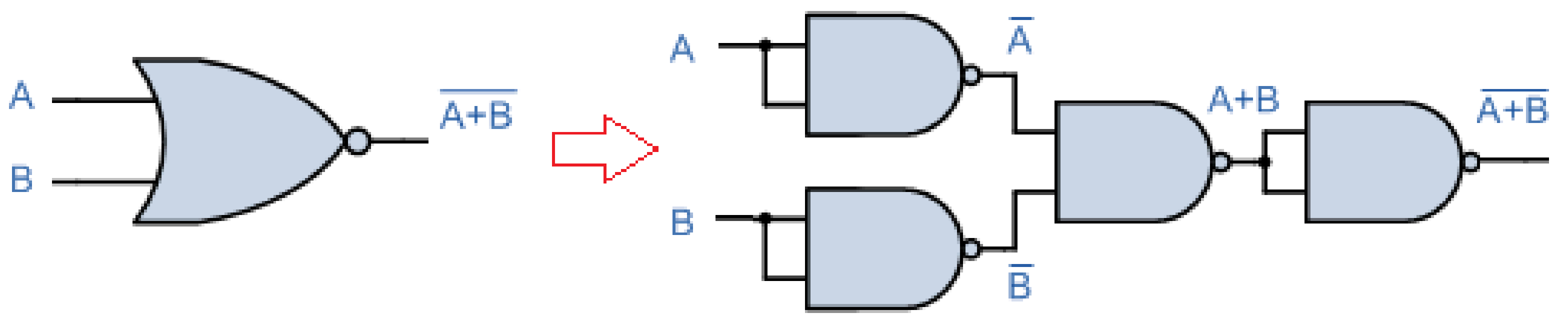
- AND Gate



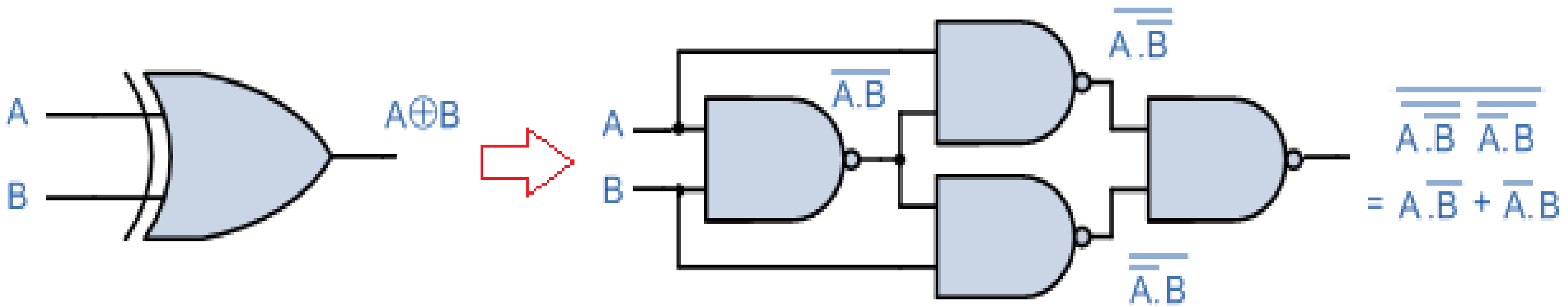
- OR Gate



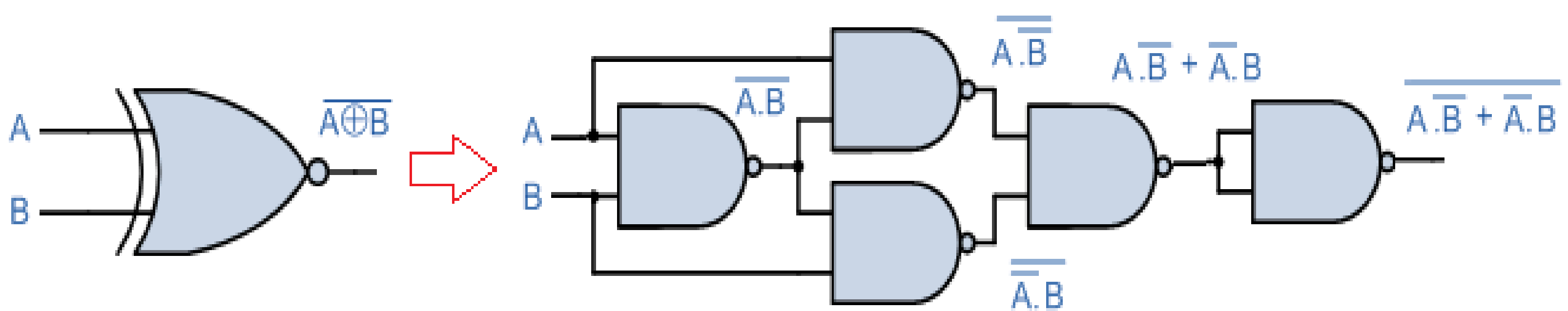
- NOR Gate



- EX-OR Gate

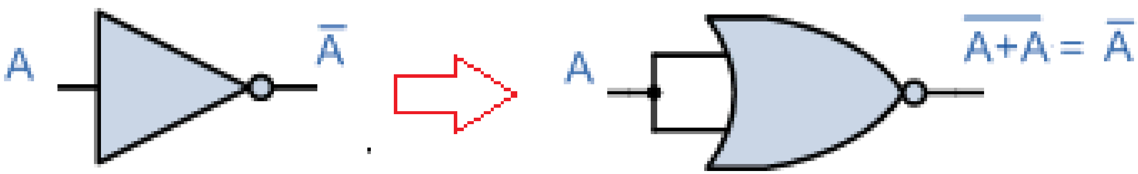


- EX-NOR Gate

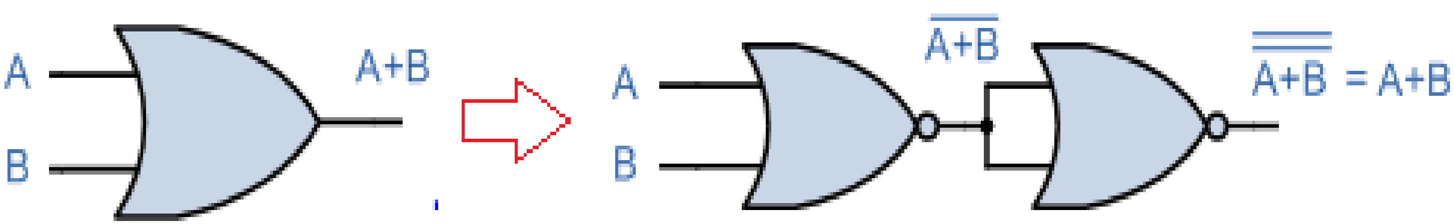


Logic Gates using only NOR Gates

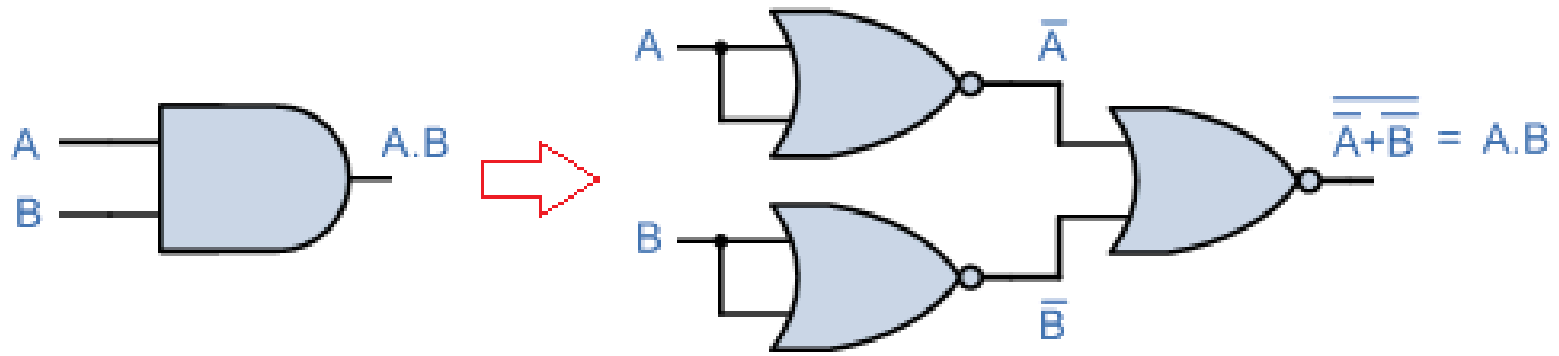
- NOT Gate



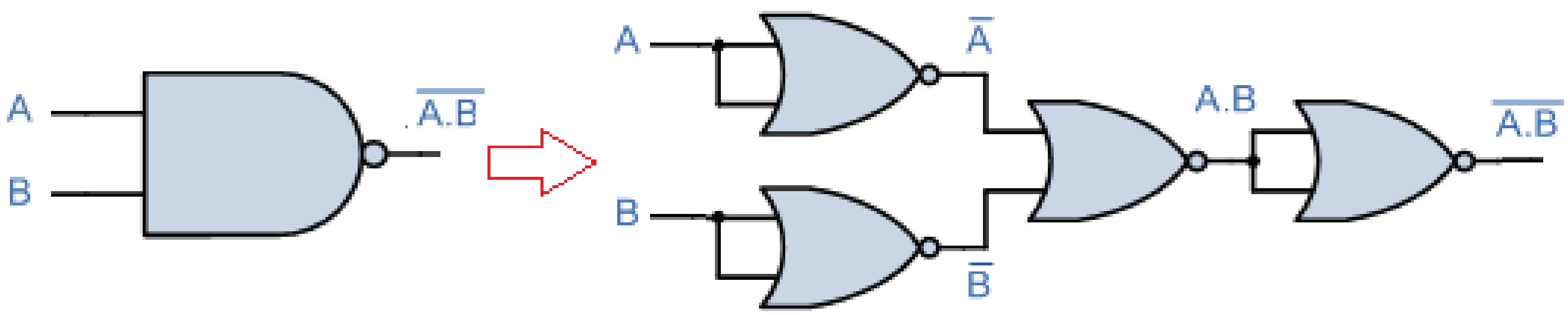
- OR Gate



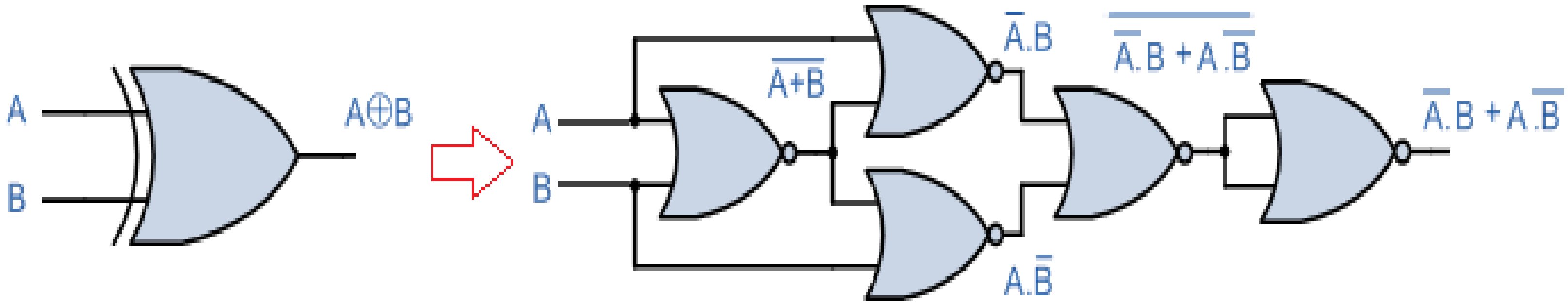
- AND Gate



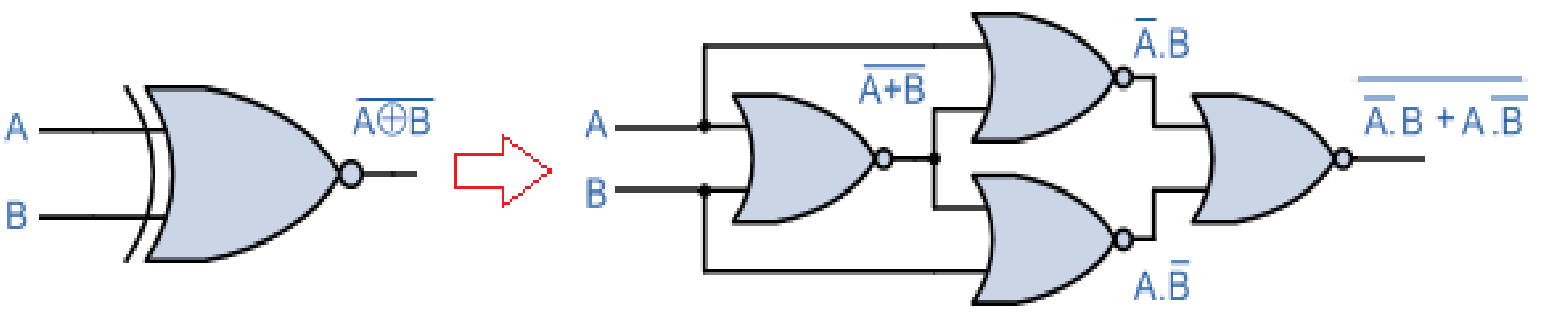
- NAND Gate



- EX-OR Gate



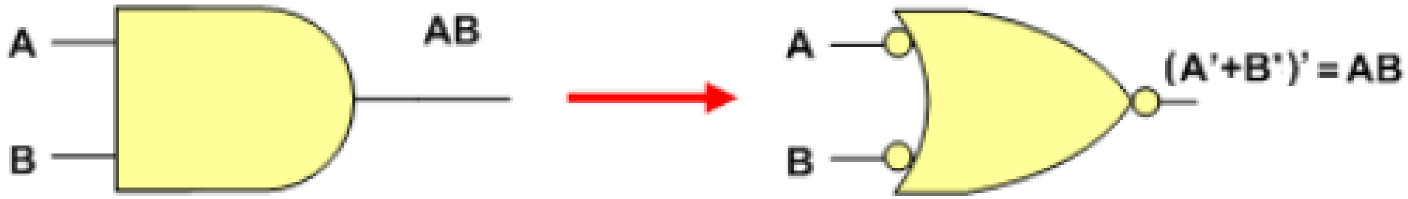
- EX-NOR Gate



Equivalent Gates

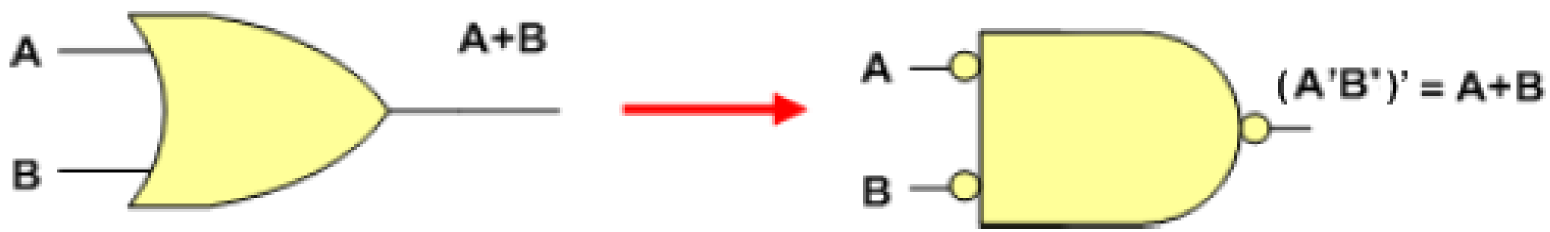
- Note that a bubble denotes complementation (inverter) and two bubbles along the same line represent double complementation, so both can be removed.
- Two NOT gates in series are same as a buffer because they cancel each other as $\bar{\bar{A}} = A$

- An AND gate is equivalent to an inverted-input NOR gate.

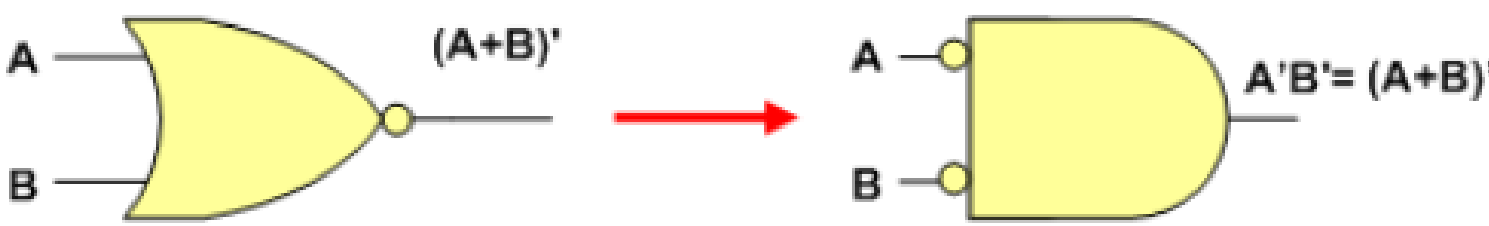


- A NAND gate is equivalent to an inverted-input OR gate.

- An OR gate is equivalent to an inverted-input NAND gate.



- A NOR gate is equivalent to an inverted-input AND gate.

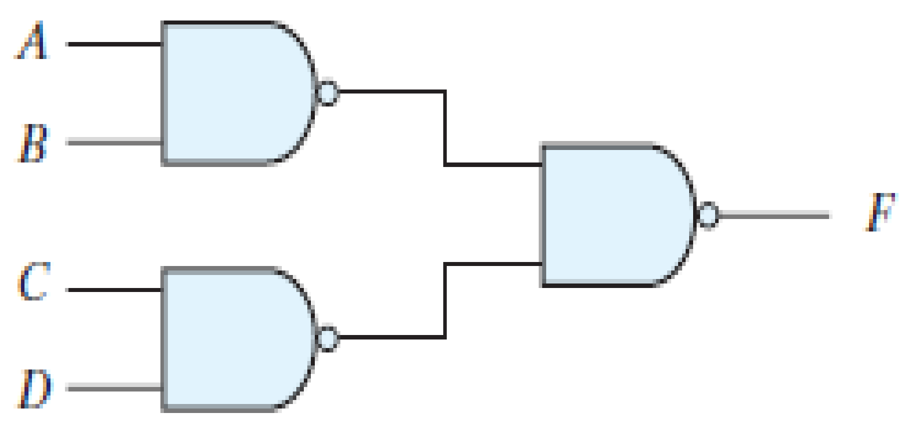


NAND Implementation

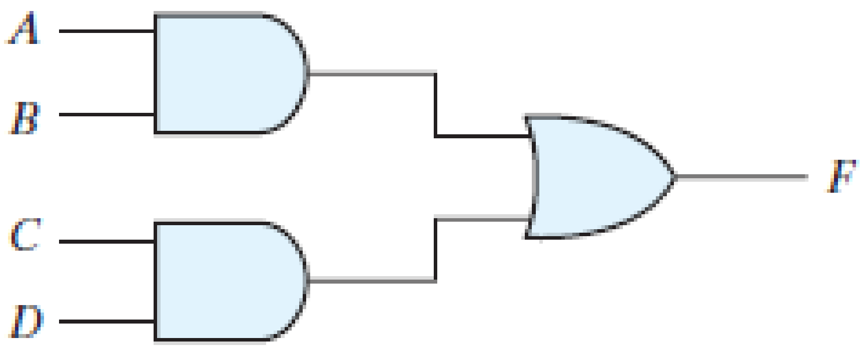
- The implementation of Boolean functions with NAND gates requires that the functions be in sum-of-products form.

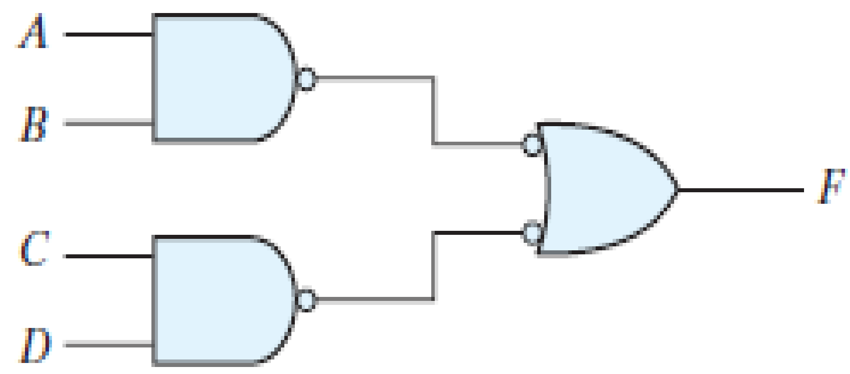
F = AB + CD

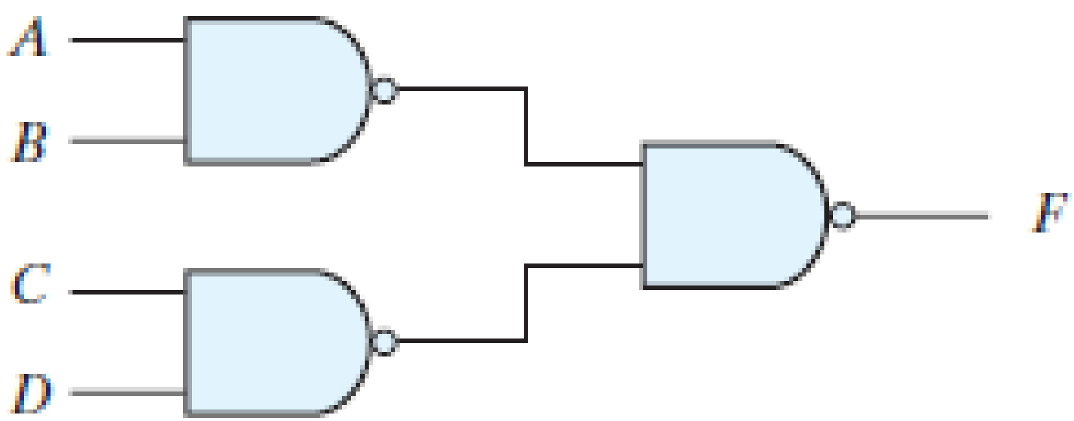
$$
\begin{aligned}
F &= AB + CD \\
&= \overline{\overline{AB + CD}} \\
&= \overline{\overline{AB}\,.\,\overline{CD}}
\end{aligned}
$$



Three ways to implement $F = AB + CD$





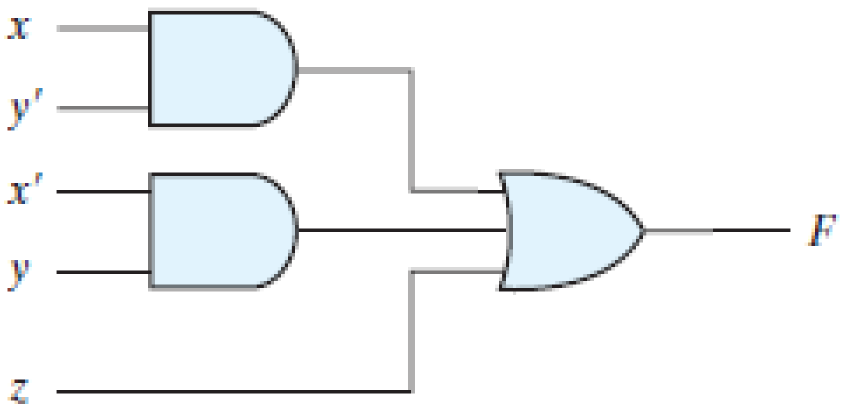


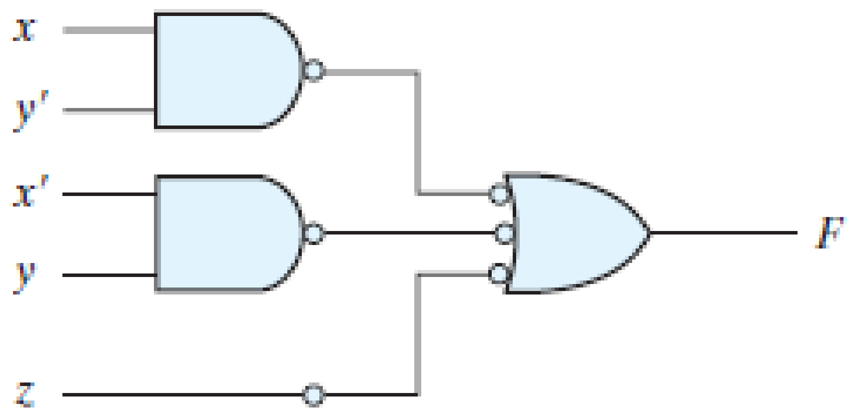
- The general procedure for implementation of a Boolean function with NAND gates using mixed notation is as follows:

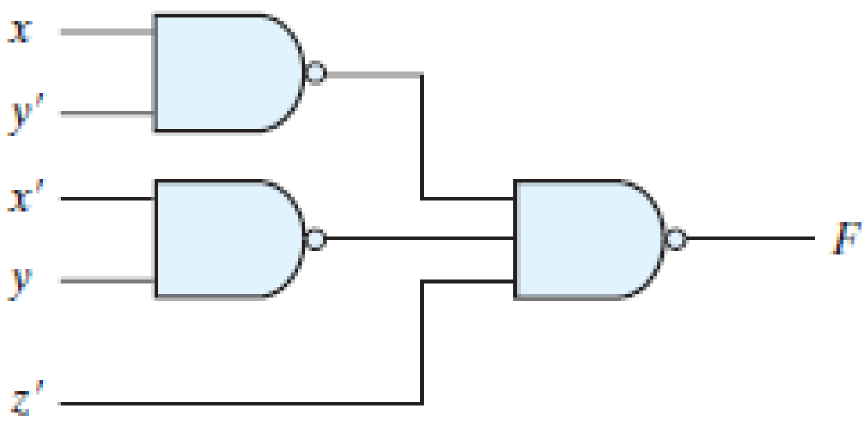
1. Draw the Boolean function by basic gates such as AND, OR and NOT gates.
2. Convert all AND gates to NAND gates with AND-invert graphic symbols.
3. Convert all OR gates to NAND gates with invert-OR graphic symbols.
4. Check all the bubbles in the diagram. For every bubble that is not compensated by another small circle along the same line, insert an inverter (a one-input NAND gate) or complement the input literal.

Example: Implement the following Boolean function with NAND gates:

$F = xy' + x'y + z$

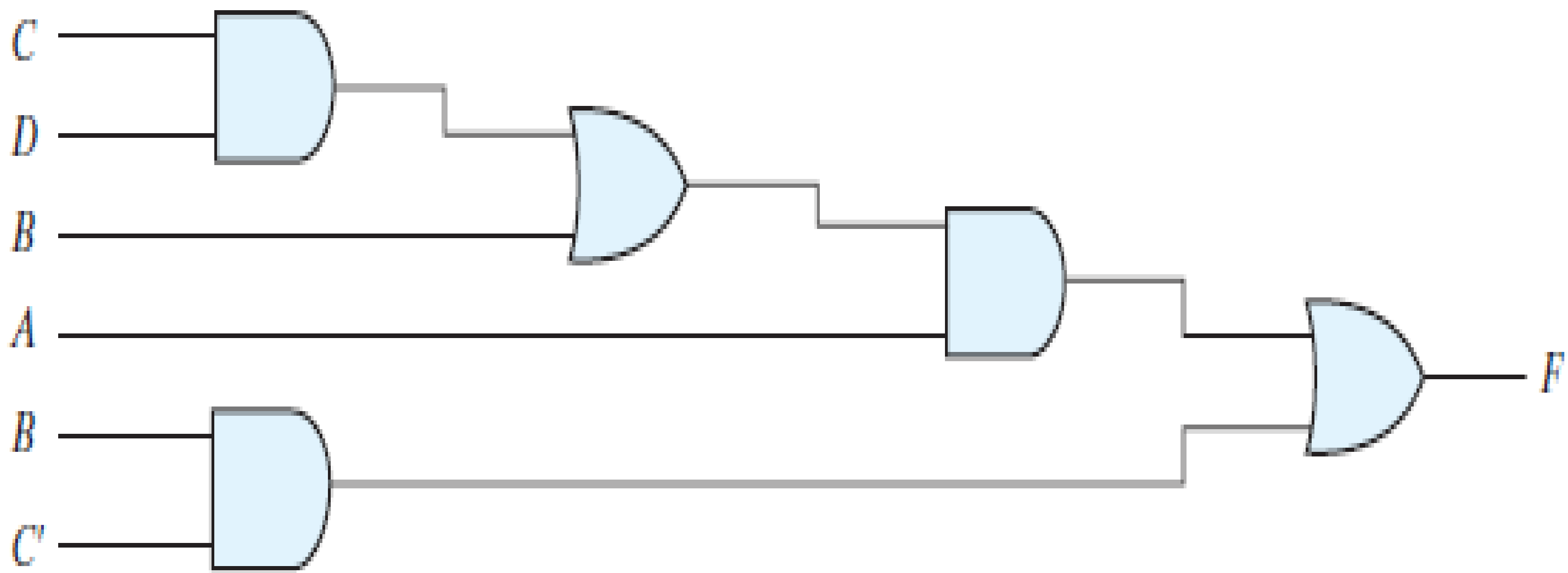


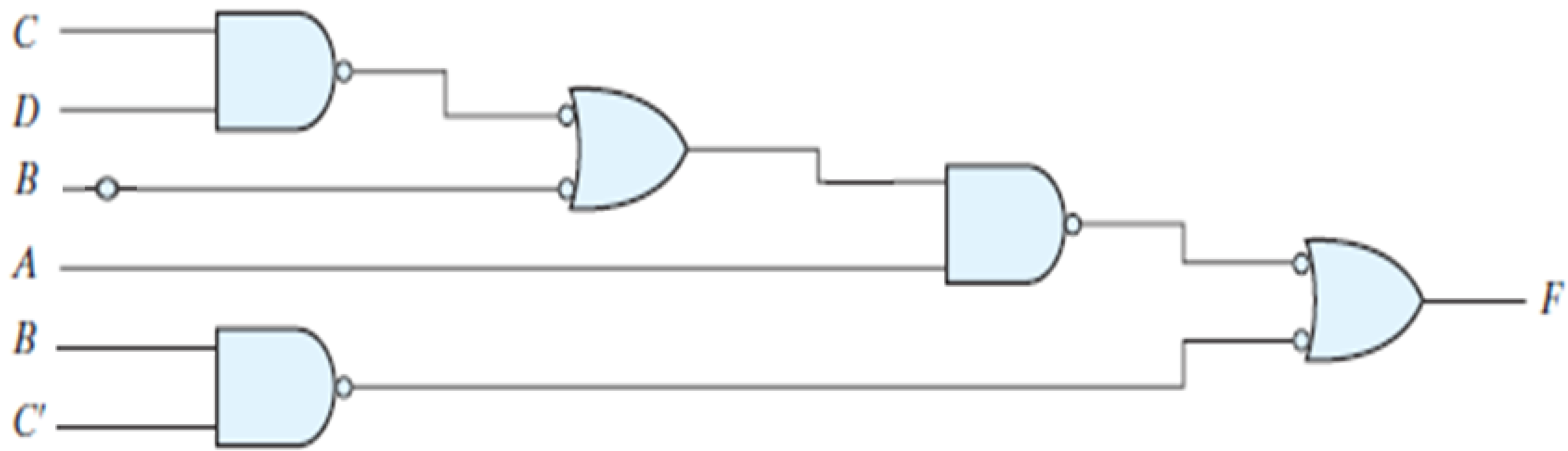


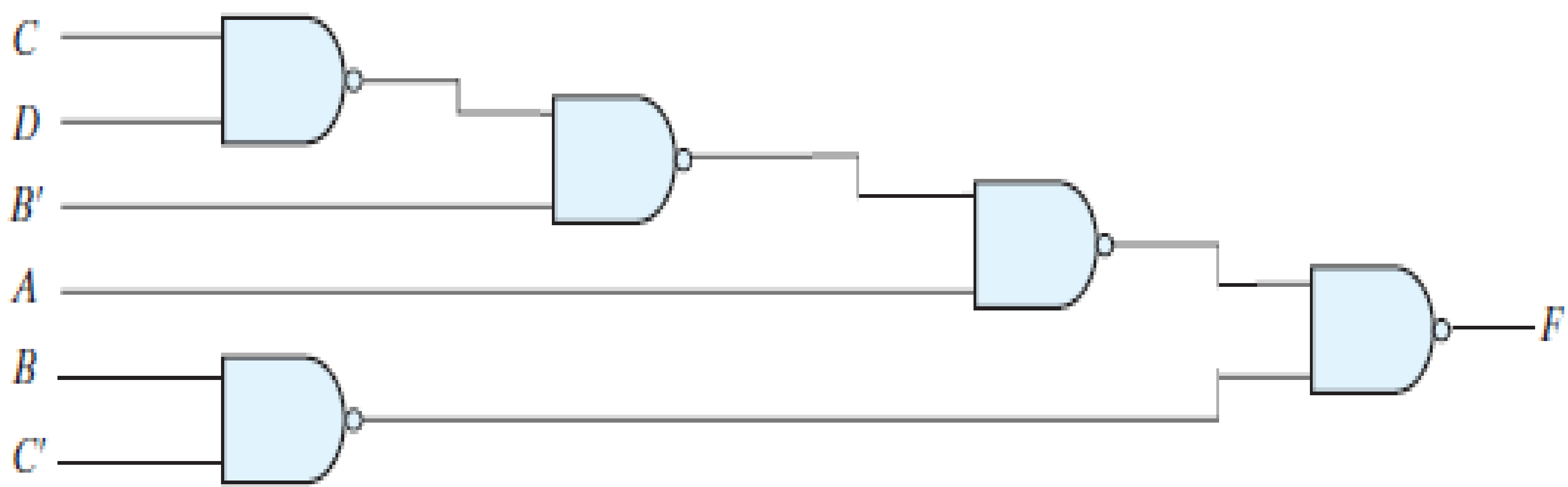


Example: **Implement the following Boolean function with NAND gates:**

$F = A(CD + B) + BC'$

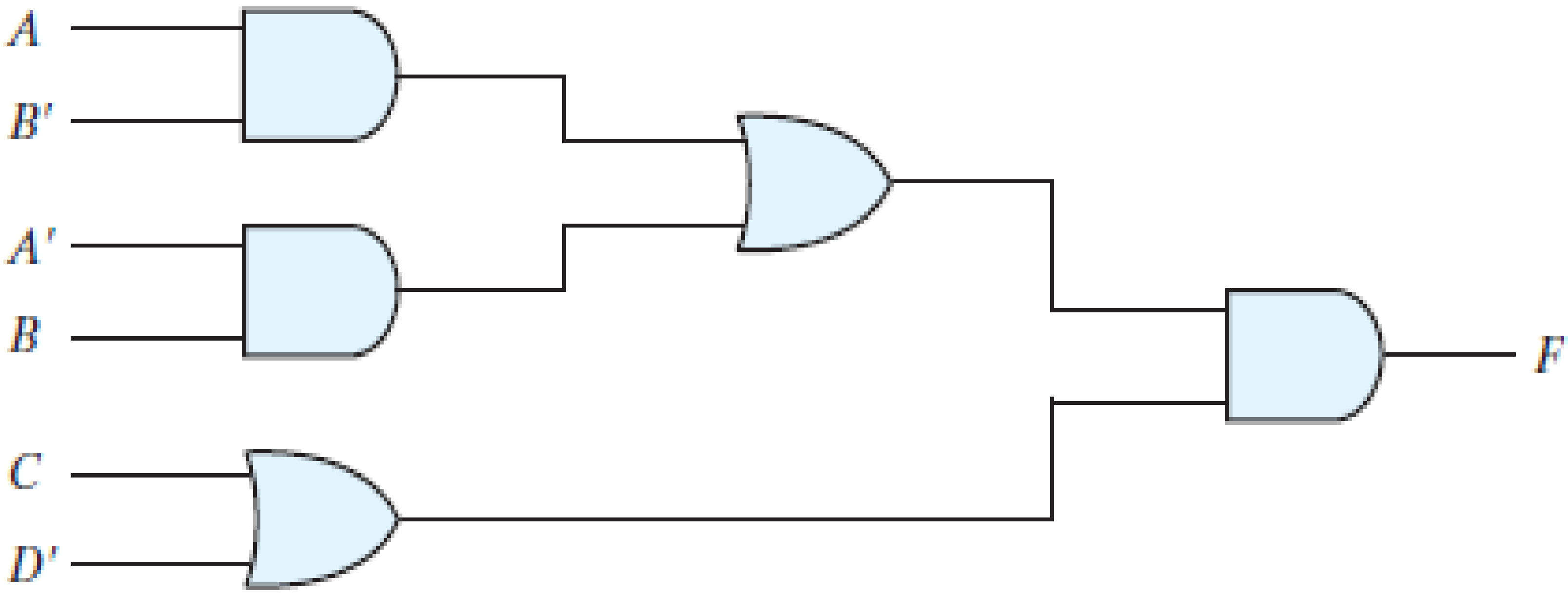


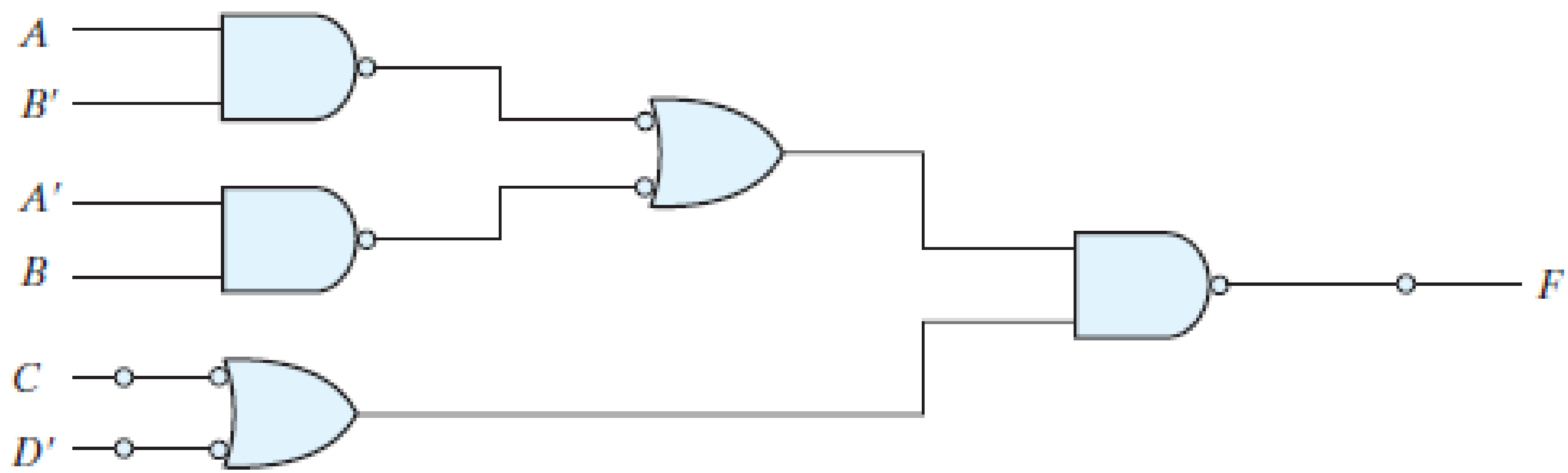

C
D
B
A
B
C'
F

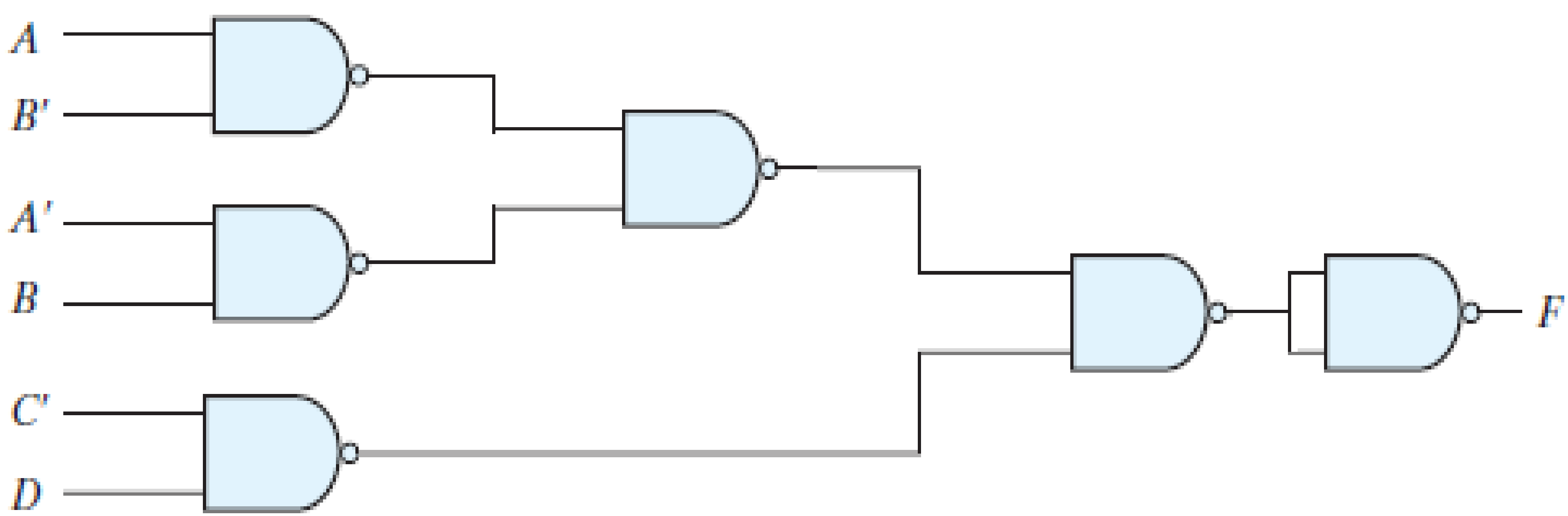

C
D
B'
A
B
C'
F

Example: **Implement the following Boolean function with NAND gates:**

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




A
B'
A'
B
C
D'
F

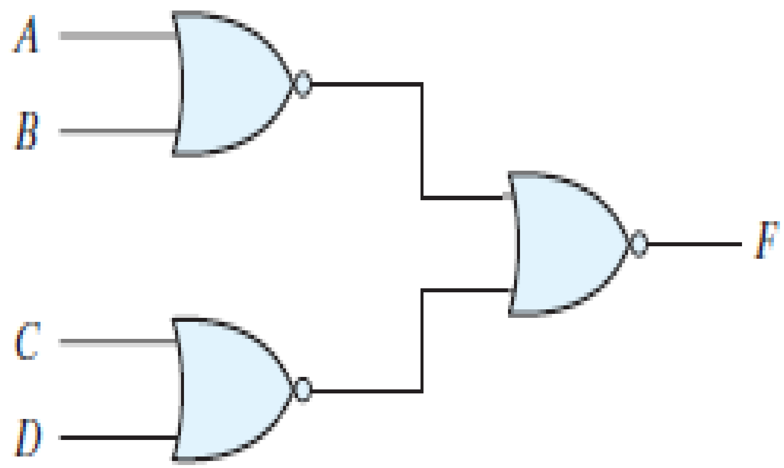

A
B'
A'
B
C'
D
F

NOR Implementation

- The implementation of Boolean functions with NOR gates requires that the functions be in product-of-sums form.

$$\boldsymbol{F = (A + B)(C + D)}$$

$$\begin{aligned} F &= (A+B).(C+D) \\ &= \overline{\overline{(A+B).(C+D)}} \\ &= \overline{\overline{(A+B)}+\overline{(C+D)}} \end{aligned}$$

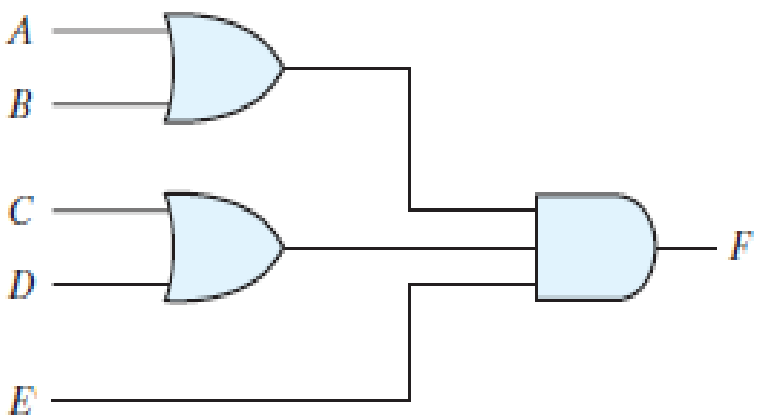


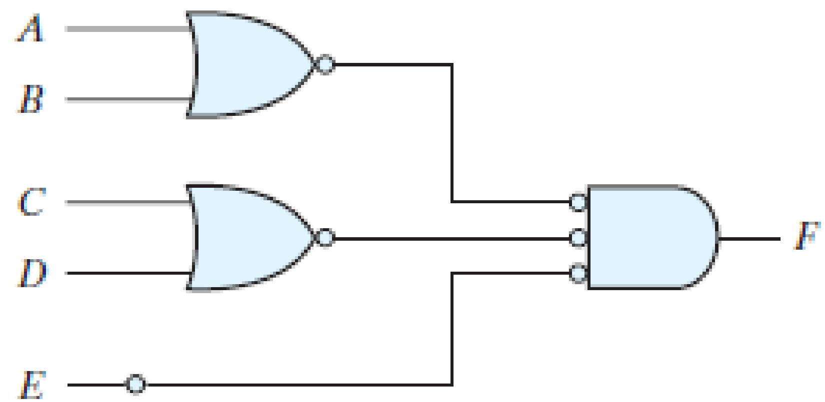
- The general procedure for implementation of a Boolean function with NOR gates using mixed notation is as follows:

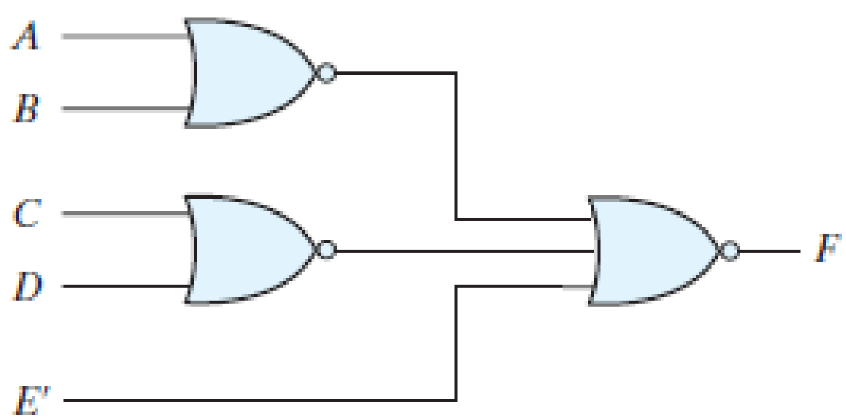
1. Draw the Boolean function by basic gates such as AND, OR and NOT gates.
2. Convert all OR gates to NOR gates with OR-invert graphic symbols.
3. Convert all AND gates to NOR gates with invert-AND graphic symbols.
4. Check all the bubbles in the diagram. For every bubble that is not compensated by another small circle along the same line, insert an inverter (a one-input NOR gate) or complement the input literal.

Example: Implement the following Boolean function with NOR gates:

$F = (A + B)(C + D)E$

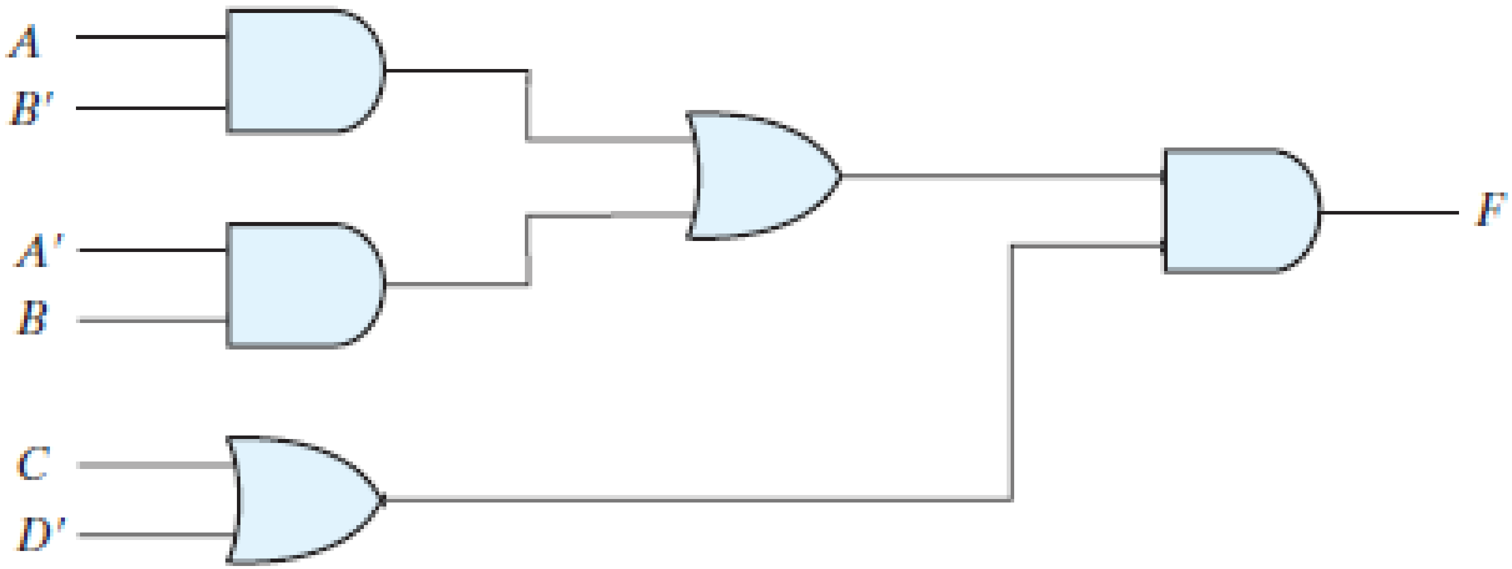


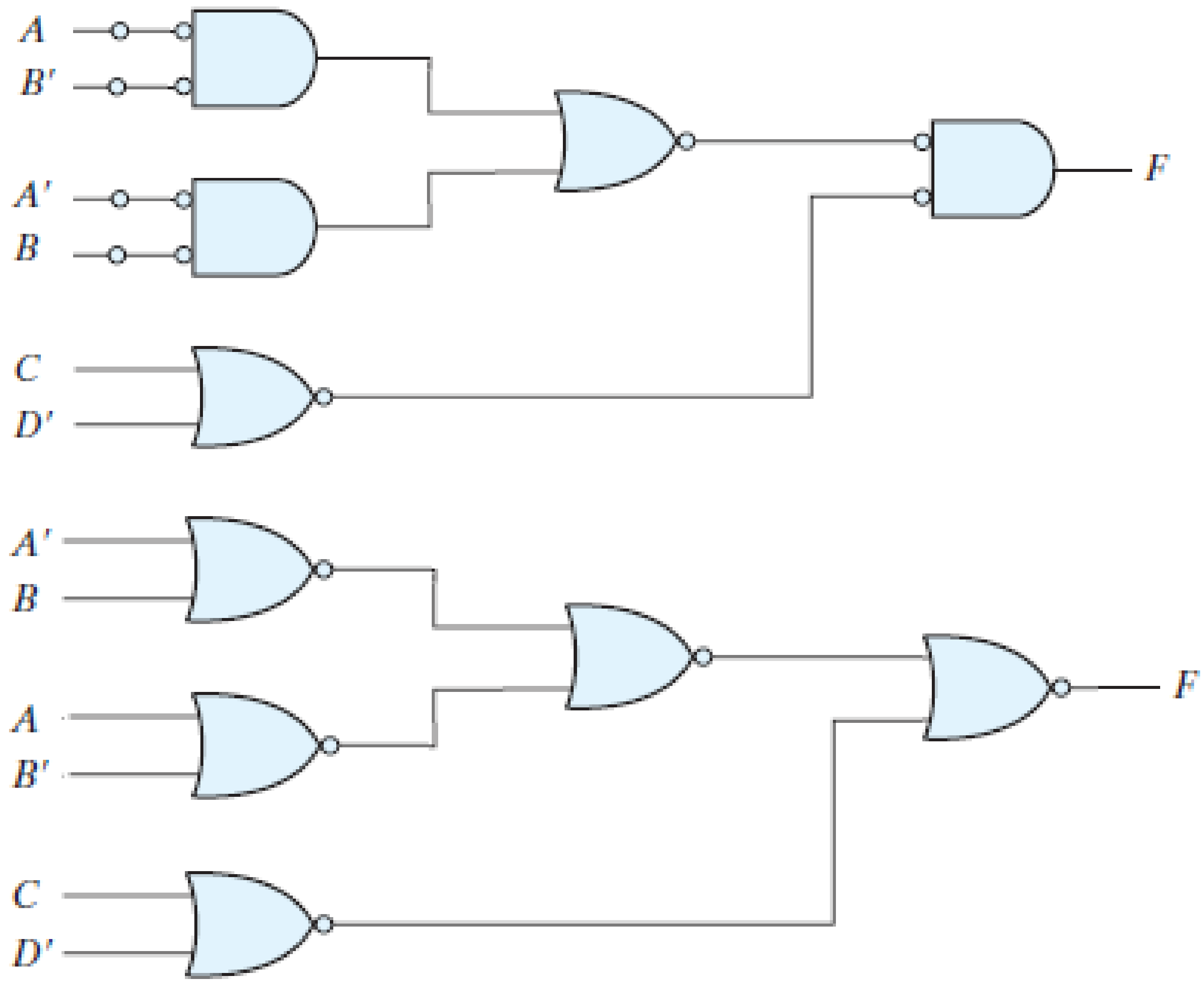




Example: **Implement the following Boolean function with NOR gates:**

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




A
B′
A′
B
C
D′
F
A′
B
A
B′
C
D′
F