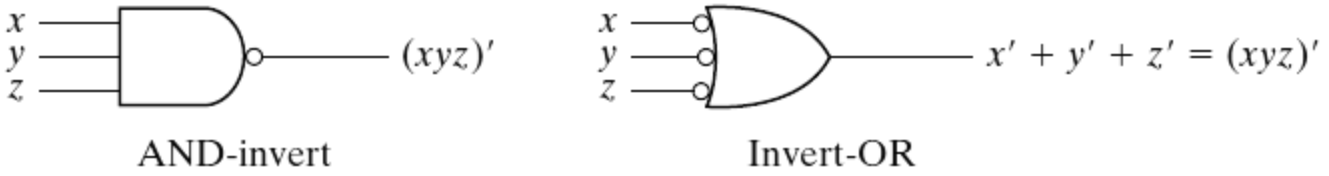
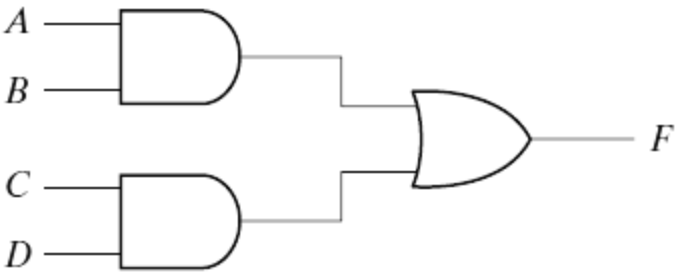
NAND Implementation

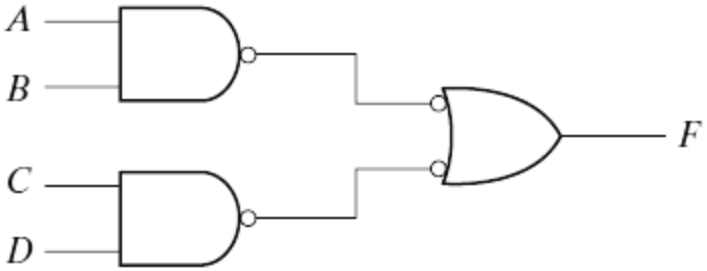
- The implementation of Boolean functions with NAND gates requires that the functions be in sum-of-products form(two-level implementation).
- The procedure for obtaining the logic diagram from a Boolean function is as follows:
  - 1. Simplify the function and express it in sum-of-products form.
  - 2. Draw a NAND gate for each product term of the expression that has at least two literals. The inputs to each NAND gate are the literals of the term. This procedure produces a group of first-level gates.
  - 3. Draw a single gate using the AND-invert or the invert-OR graphic symbol in the second level, with inputs coming from outputs of first-level gates.
  - 4. A term with a single literal requires an inverter in the first level. However, if the single literal is complemented, it can be connected directly to an input of the second level NAND gate.

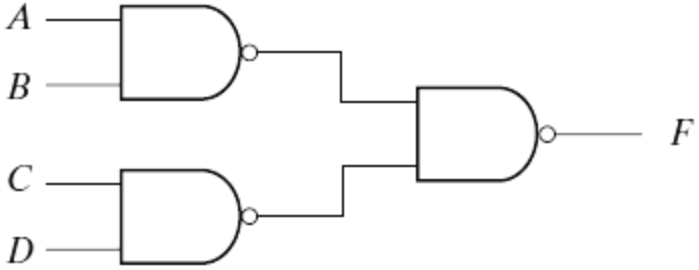


NAND Implementation

- $F = AB + CD$

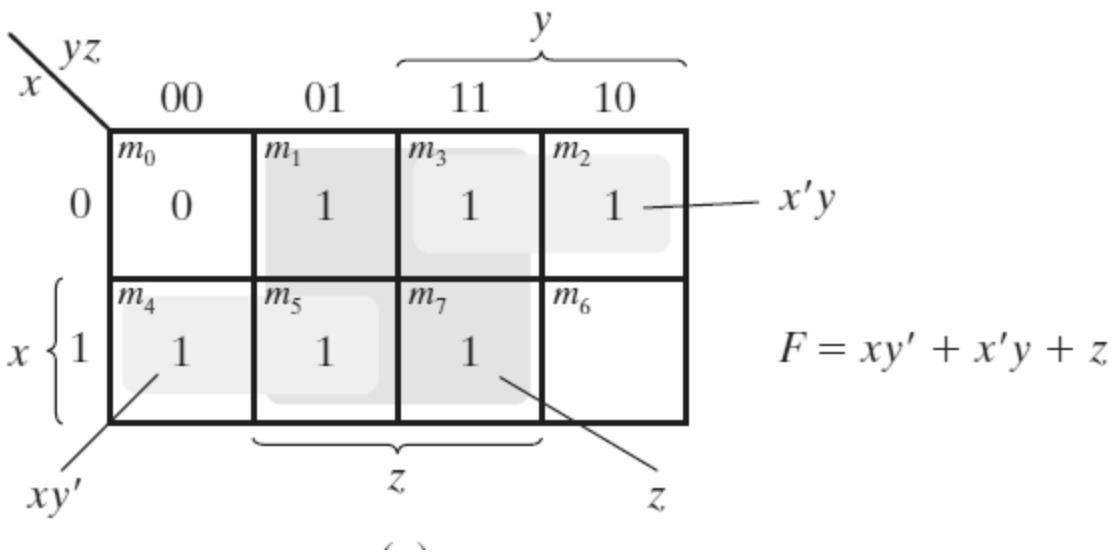






NAND Implementation

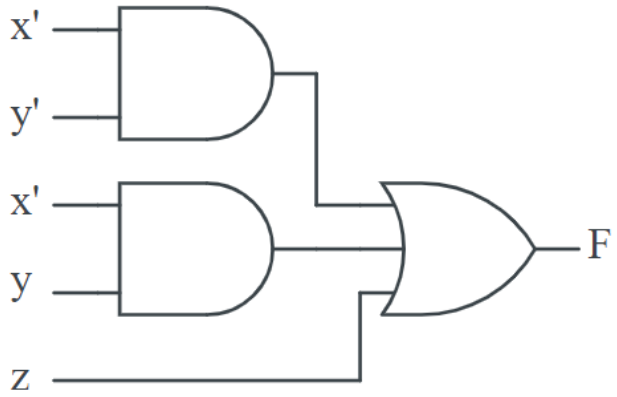
- Implement the following Boolean function with NAND gates:
  - *F (x, y, z) = ∑(1, 2, 3, 4, 5, 7)*
    - Simplified SOP:



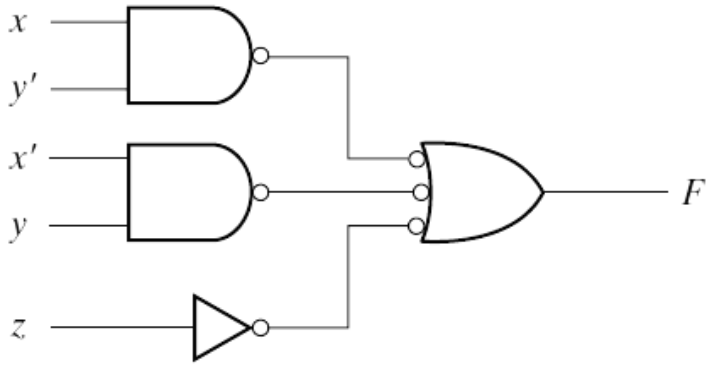
NAND Implementation

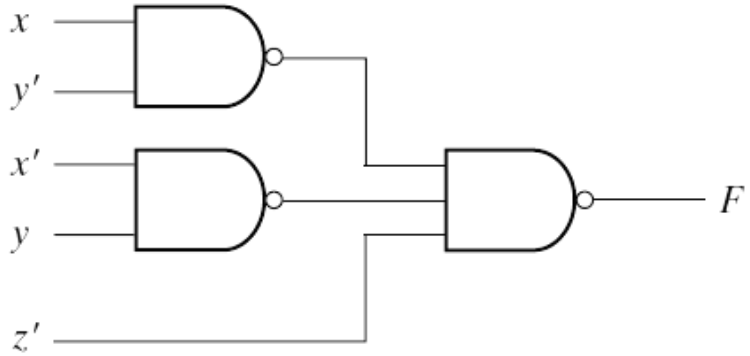
$$F = xy' + x'y + z$$

- Two-level implementation:



- NAND implementation:





NAND Implementation

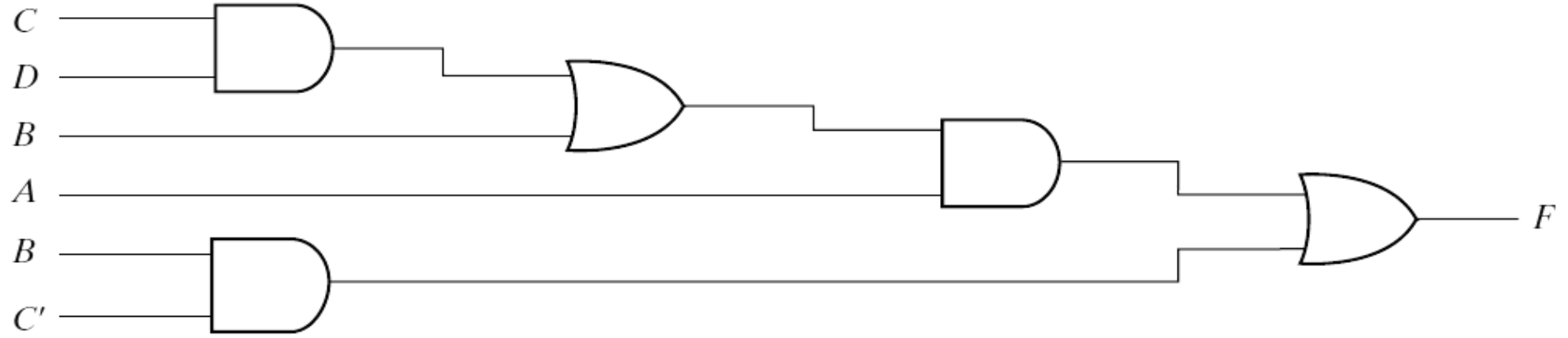
- **Multilevel NAND Circuits:**
  - The standard form of expressing Boolean functions results in a two-level implementation.
  - There are occasions, however, when the design of digital systems results in gating structures with three or more levels.
  - The most common procedure in the design of multilevel circuits is to express the Boolean function in terms of AND, OR, and complement operations.

NAND Implementation

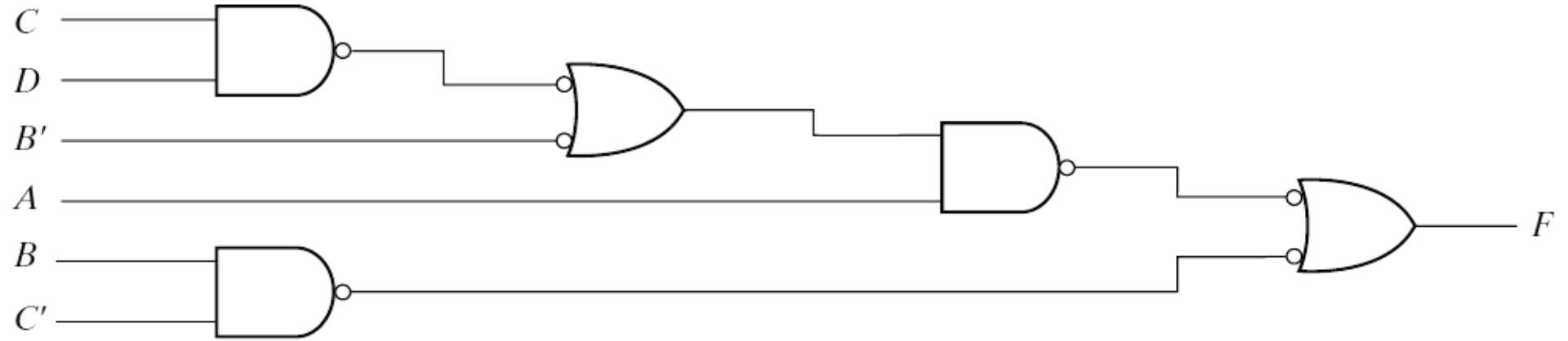
- The general procedure for converting a multilevel AND–OR diagram into an all-NAND diagram using mixed notation is as follows:
  - 1. Convert all AND gates to NAND gates with AND-invert graphic symbols.
  - 2. Convert all OR gates to NAND gates with invert-OR graphic symbols.
  - 3. Check all the bubbles 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.

NAND Implementation

- Consider the Boolean function
  - *F = A (CD + B) + BC'*



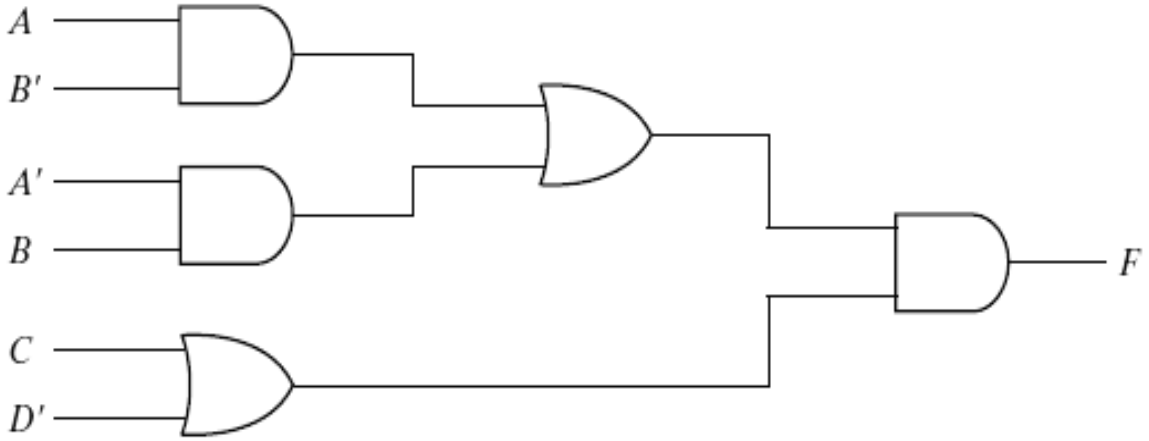
AND–OR gates

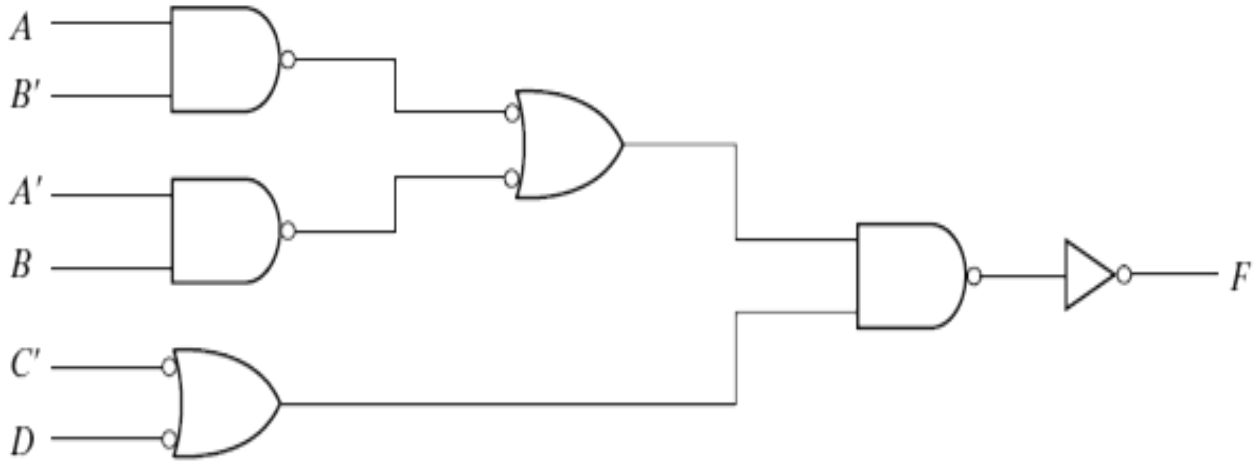


NAND gates

NAND Implementation

- Consider the multilevel Boolean function
  - *F = (AB‘ + A‘B)(C + D‘)*



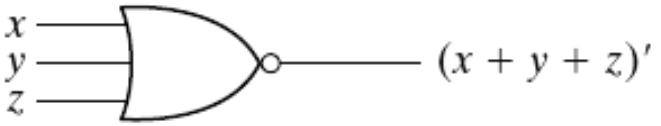


NOR Implementation

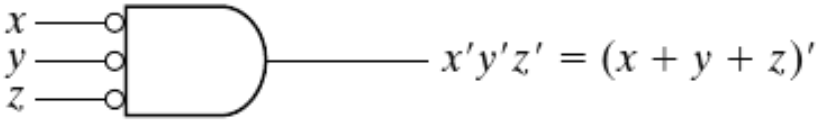
- The NOR operation is the dual of the NAND operation.
  - Therefore, all procedures and rules for NOR logic are the duals of the corresponding procedures and rules developed for NAND logic.
- The NOR gate is another universal gate that can be used to implement any Boolean function.

NOR Implementation

- A two-level implementation with NOR gates requires that the function be simplified into product-of-sums form.



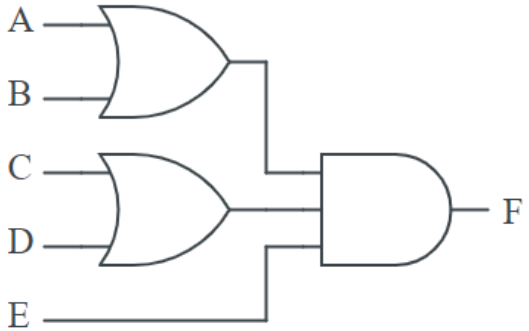
OR-invert

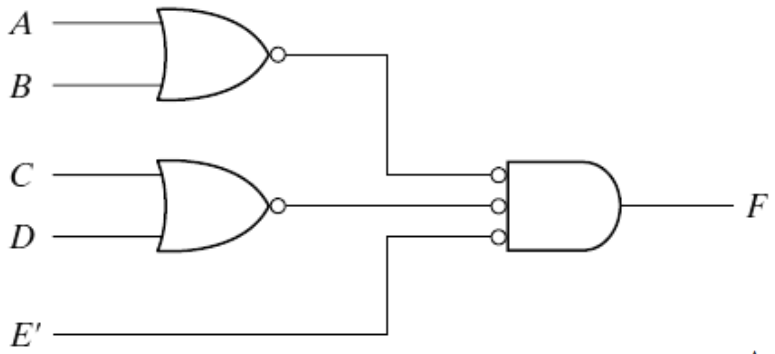


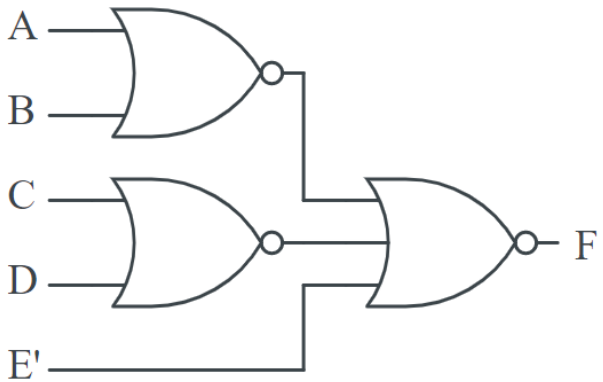
Invert-AND

NOR Implementation

- *F = (A + B)(C + D)E*

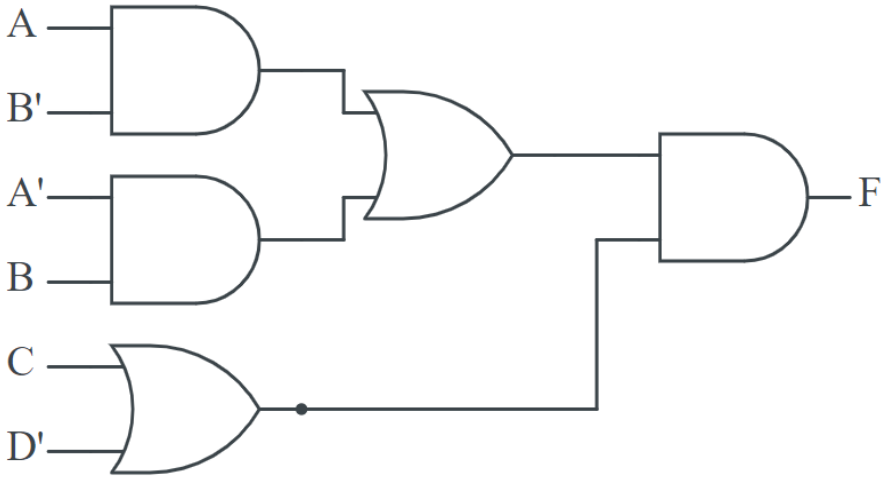


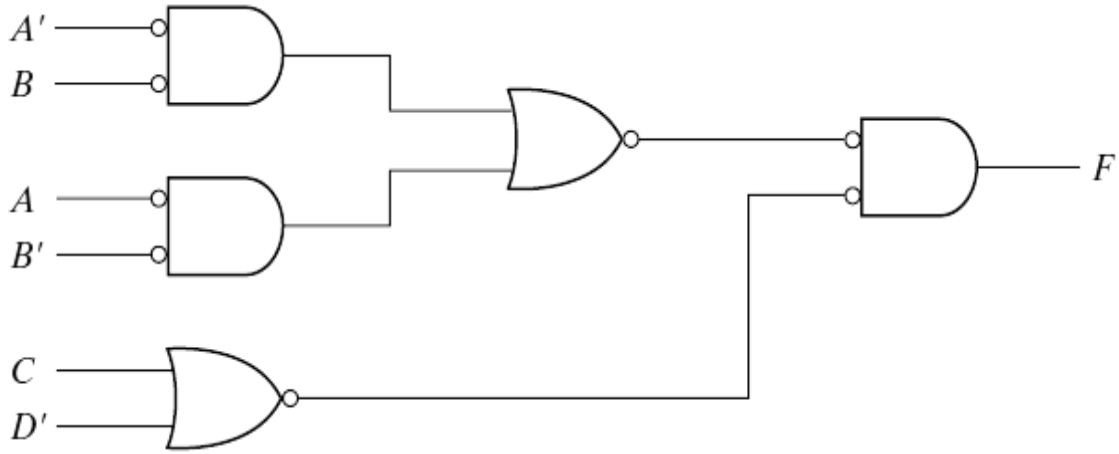




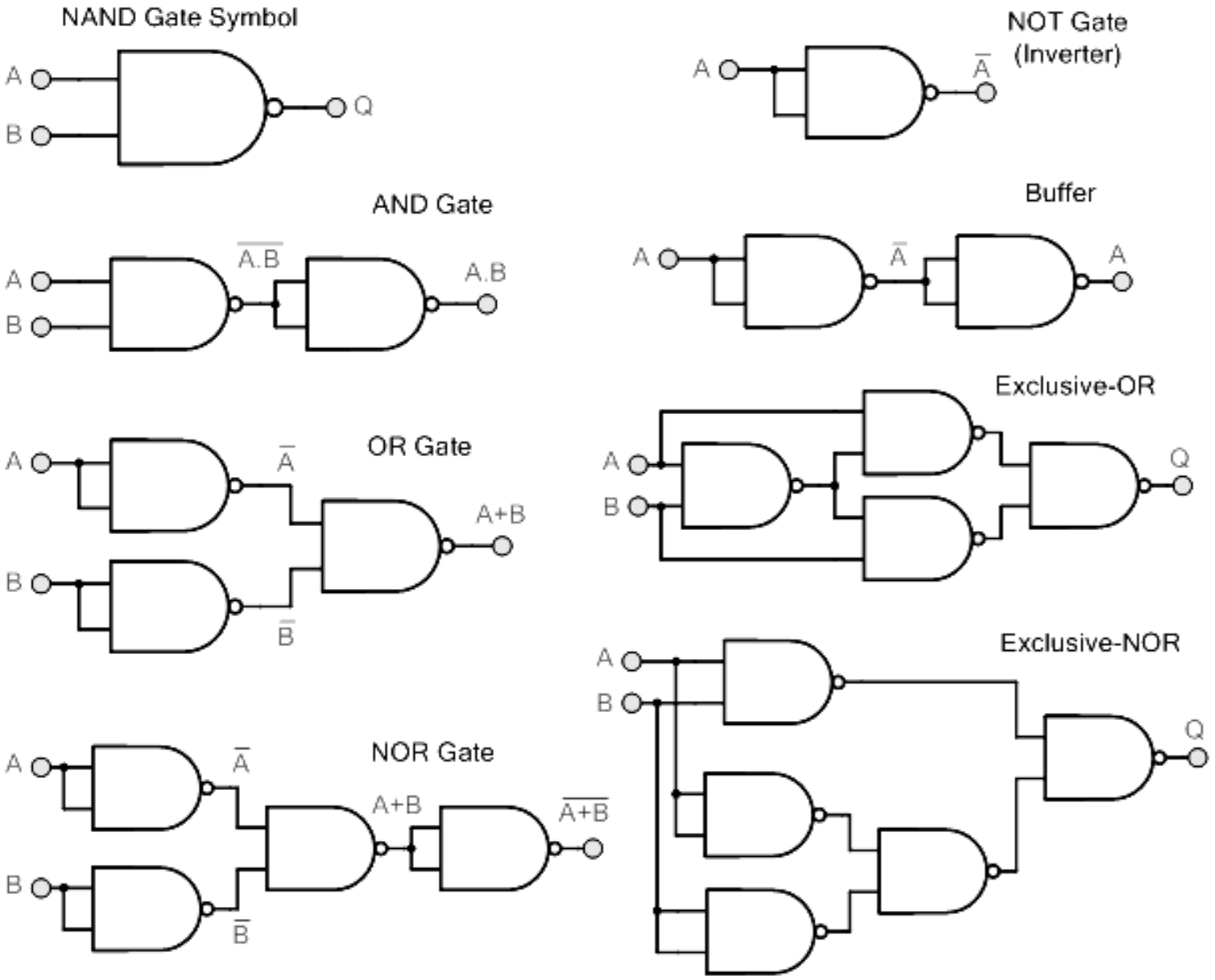
NOR Implementation

- Consider the multilevel Boolean function
  - *F = (AB‘ + A‘B)(C + D‘)*





Logic gates using NAND



Logic gates using NOR