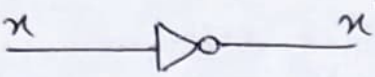
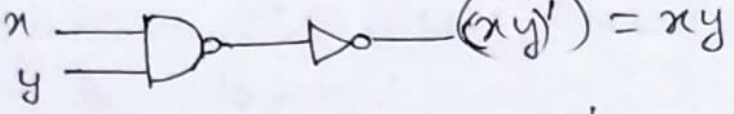
- NAND and NOR Implementations (Multilevel NAND/NOR Realizations)

→ The NAND gate is called universal gate because any logic circuit can be implemented with it.

→ To show that any Boolean function can be implemented with NAND gates, we need to show that the logical operations of AND, OR and complement (~~oper~~) can be obtained with NAND gates alone.

Inverter 

AND 

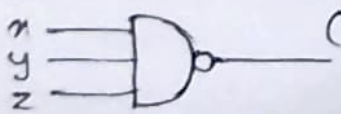
OR 

→ A convenient way to implement a Boolean function with NAND gates is to obtain the simplified Boolean function in terms of Boolean operators and then convert the function to NAND logic.

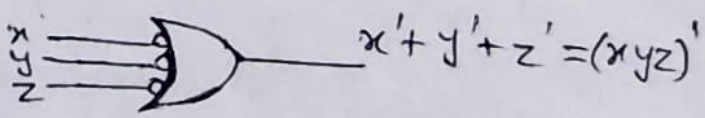
→ The conversion of an algebraic expression from AND, OR, and complement to NAND can be done by simple circuit manipulation techniques that change AND-OR diagrams to NAND diagrams.

→ To facilitate the conversion to NAND logic, it is convenient to define an alternative graphic symbols for the gate.

→ Two equivalent graphic symbols for the NAND gate are as follows:



AND-Invert



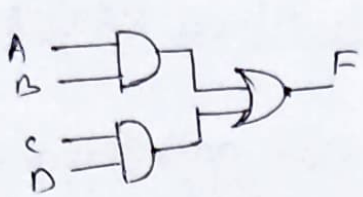
Invert-OR

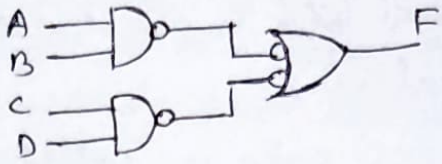
→ The two graphic symbols representations are useful in the analysis and design of NAND circuits.

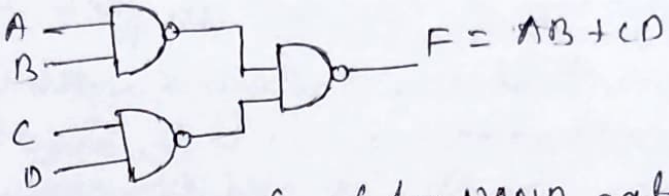
Two-level Implementation

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

→ Consider the function $F = AB + CD$







→ The AND gates are replaced by NAND gates and the OR gate is replaced by a NAND gate with an Invert-OR graphic symbol.

→ Then the output NAND gate is redrawn with the AND-invert graphic symbol.

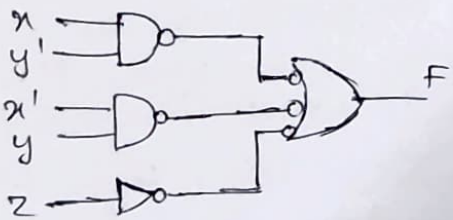
→ The NAND implementation can be verified algebraically.

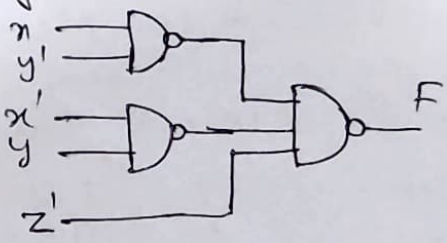
$$F = ((AB)'(CD)')' = AB + CD$$

Q) Implement the following function with NAND gates:

$$F(x, y, z) = \sum(1, 2, 3, 4, 5, 7)$$

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





→ 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 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.

Multilevel NAND Circuits

→ The standard form of expressing Boolean functions results in a two-level implementation.

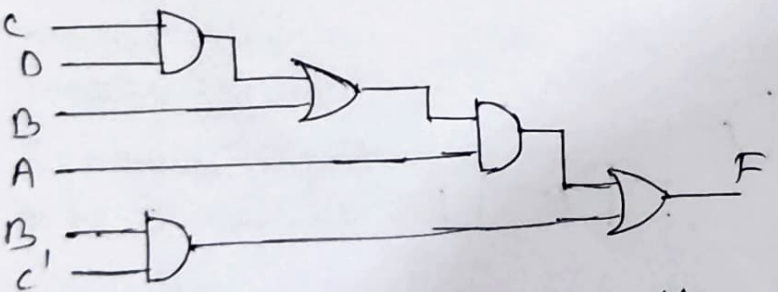
→ There are occasions, when the design of digital systems results in gating structures with three or more levels.

→ The most common procedure in the design of multilevel circuits is to express the Boolean function in terms of AND, OR, and complement operations. The function can then be implemented with AND and OR gates

→ After that it can be converted into all NAND circuit.

→ Consider the Boolean function $F = A(CD+B)+BC'$

→ Although it is possible to remove parentheses and reduce the expression into a standard sum-of-products form, we take the exact expression to implement it as a multilevel circuit



→ A logic diagram with a pattern of alternating levels of AND and OR gates can easily be converted into a NAND circuit.

→ The procedure is to change every AND gate to an AND-invert graphic symbol and every OR gate to an invert-OR graphic symbol.

→ The general procedure for converting a multilevel AND-OR diagram into an all-NAND diagram 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.

Q→ Consider the multilevel Boolean function

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