NAND and NOR Implementation

Lecture-14

By
Bhagyalaxmi Behera
Asst. Professor (Dept. of ECE)

Contents

Introduction

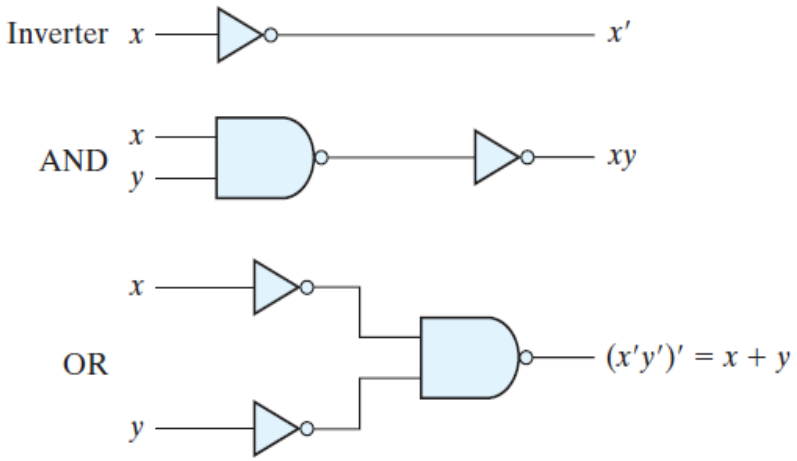
- Digital circuits are frequently constructed with **NAND** or **NOR** gates rather than with AND and OR gates.

- NAND and NOR gates are easier to fabricate with electronic components and are the basic gates used in all IC digital logic families.

- Because of the prominence of NAND and NOR gates in the design of digital circuits, rules and procedures have been developed for the conversion from Boolean functions given in terms of AND, OR, and NOT into equivalent NAND and NOR logic diagrams.

NAND Circuits

- The NAND gate is said to be a ***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 only show that the logical operations of AND, OR, and complement can be obtained with NAND gates alone.(As shown below)

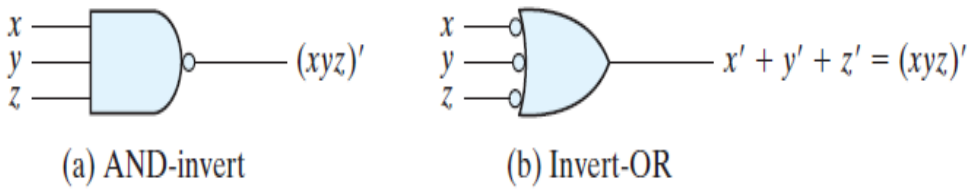


Logic operations with NAND gates

Three-input NAND gate

- 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.

$$(xyz)' = x' + y' + z'$$



Two graphic symbols for a three-input NAND gate

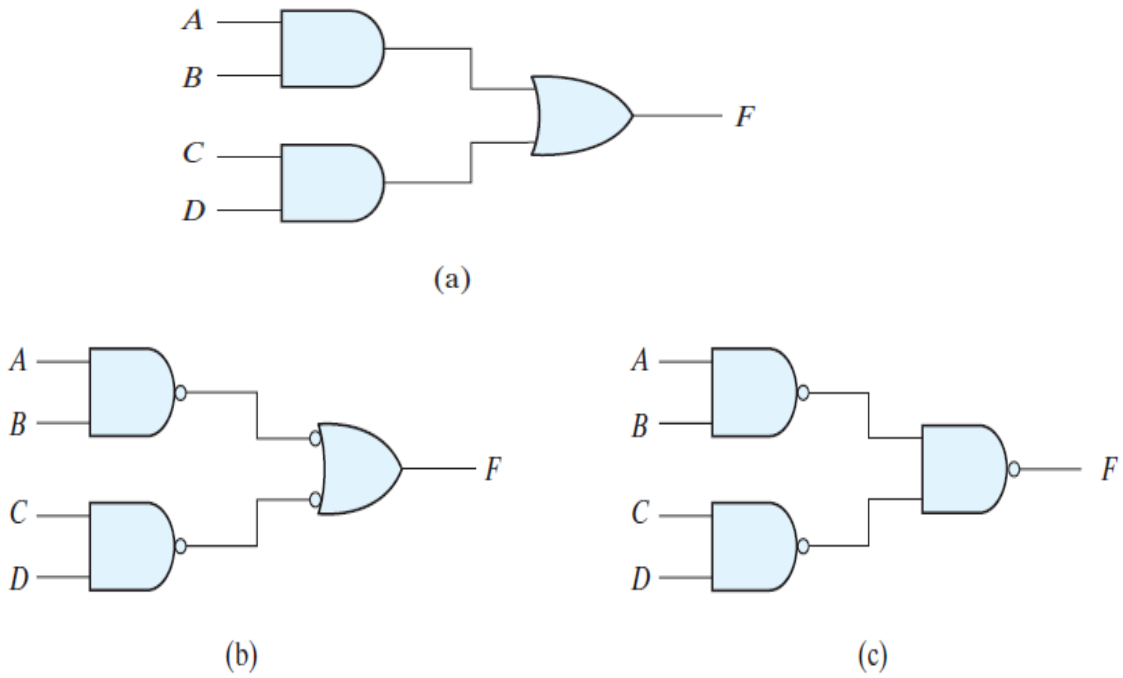
Two-Level Implementation

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

Example 1

F = AB + CD

F = ((AB + CD) ’) ’ = ((AB) ’(CD) ’) ’

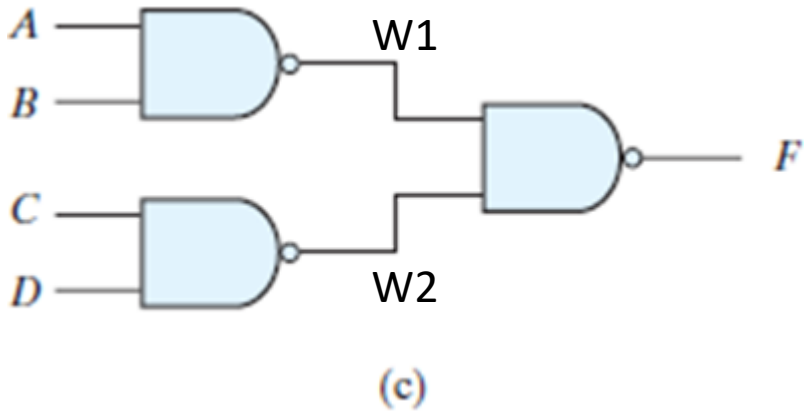


Three ways to implement *F = AB + CD*

HDL for the above circuit:

// Verilog model of two level
implementation circuit using NAND gates.

```
module  two-level-gf (A, B, C, D, F);
output F;
input A, B, C, D;
wire w1, w2;
nand G1 (w1, A, B);
nand G2 (w2, C, D);
nand G3 (F, w1, w2);
endmodule
```



(c)

Example 2

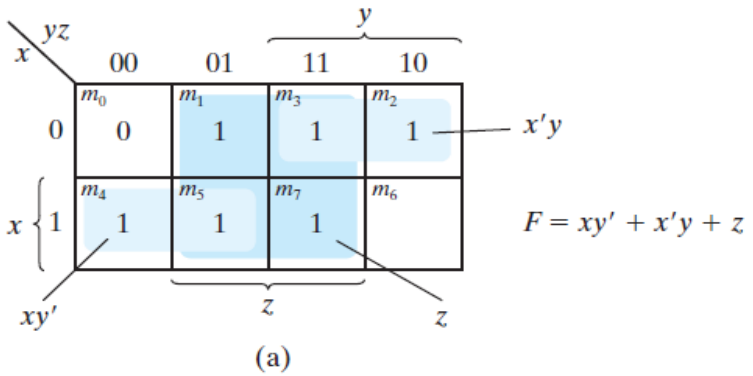
Implement the following Boolean function with NAND gates: $F(x, y, z) = \sum(1, 2, 3, 4, 5, 7)$

The procedure described in the previous example indicates that a Boolean function can be implemented with two levels of NAND gates. The procedure for obtaining the logic diagram from a Boolean function is as follows:

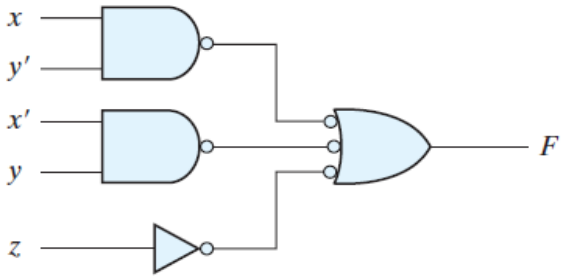
1. Simplify the function and express it in sum-of-products form.
2. Draw a NAND gate for each product term of the expression that has at least two literals. The inputs to each NAND gate are the literals of the term. This procedure produces a group of first-level gates.
3. Draw a single gate using the 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.

Answer

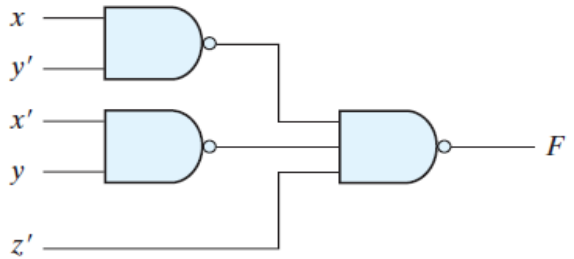
F (x, y, z) =∑ (1, 2, 3, 4, 5, 7)



(a)



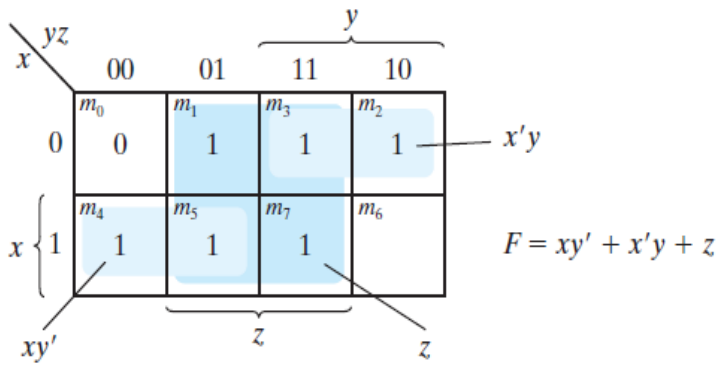
(b)

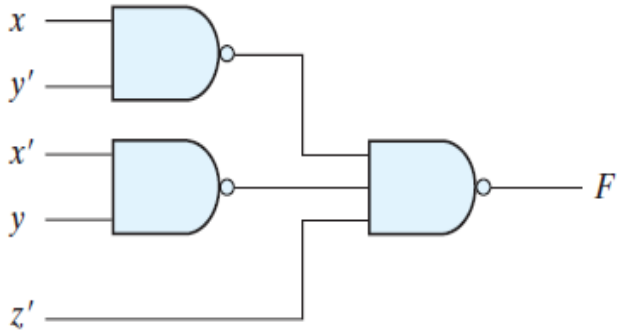


(c)

HDL for the above Boolean function

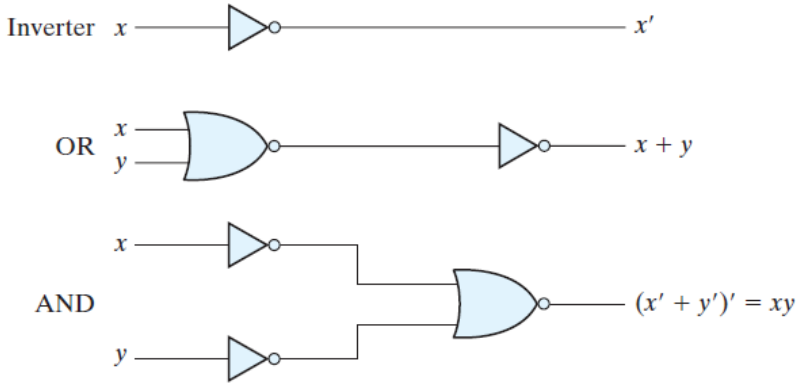
```
// Verilog model: Circuit with Boolean
expressions
module Circuit_Boolean_df (F, x, y, z);
output  F;
input x, y, z;
assign F = (x && (!y)) || ((!x) && y) ||  z ;
endmodule
```



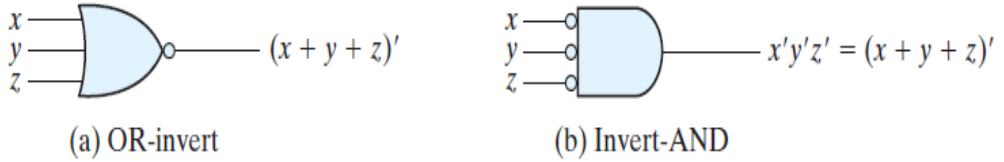


NOR Implementation

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.



Logic operations with NOR gates

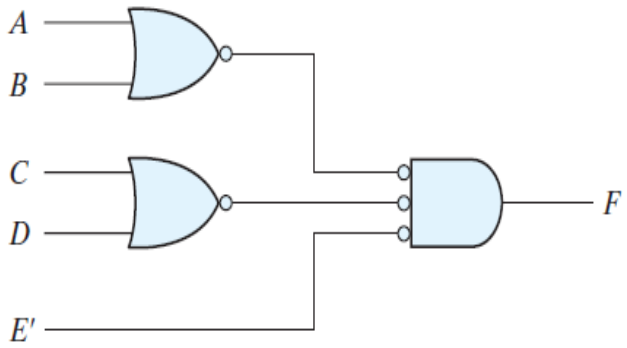


Two graphic symbols for the NOR gate

Example1

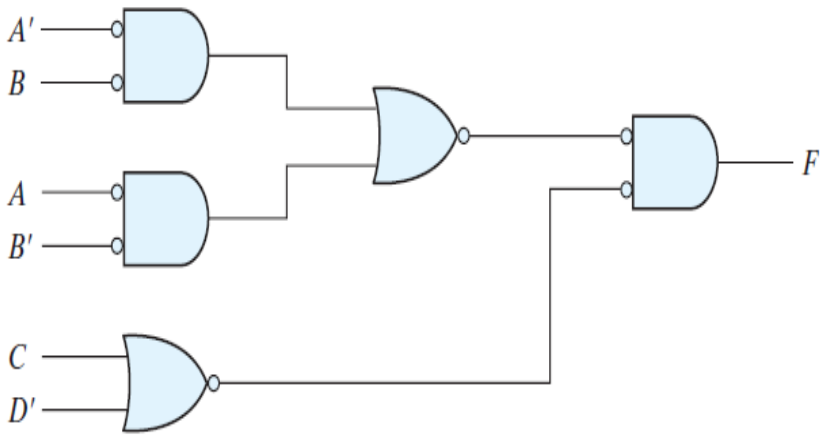
F = (A + B)(C + D)E

The OR–AND pattern can easily be detected by the removal of the bubbles along the same line. Variable *E is complemented to compensate for the third bubble at the input* of the second-level gate.



Example 2

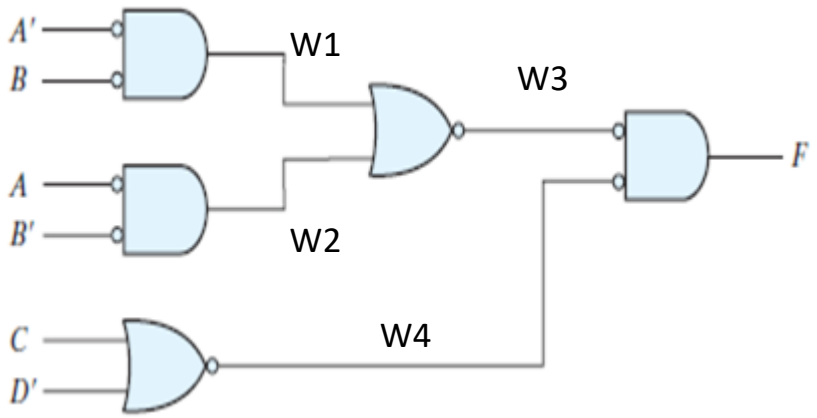
$F = (AB' + A'B)(C + D')$



HDL for the above Boolean function using NOR gates

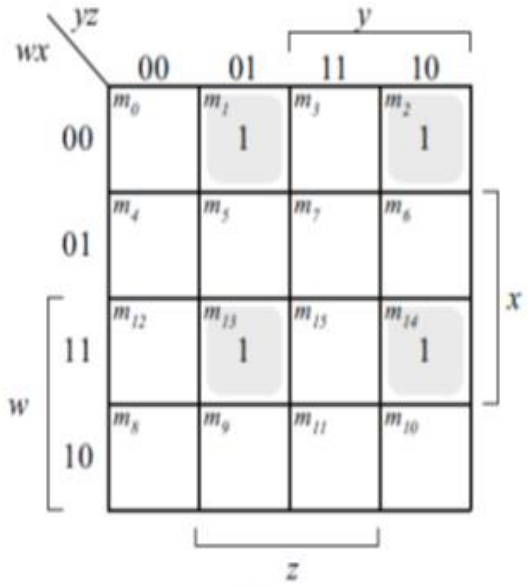
$F = (AB' + A'B)(C + D')$

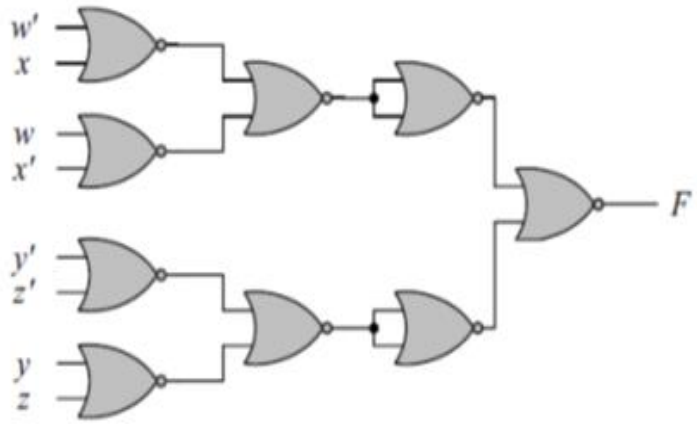
```
// Verilog model of Boolean function using
NOR gates.
module  bolean-usingnor-gf (F, A, B, C,
D);
output  F;
input A, B, C, D;
wire A', B', D', w1, w2, w3, w4;
nor  (A', A, A),
      (B', B, B),
      (D', D, D);
nor  G1 (w1, A', B),
       G2 (w2, A, B'),
       G3 (w4, C, D'),
       G4 (w3, w1, w2),
       G5 (F, w3, w4);
endmodule
```



Example 3

Implement the following Boolean function with NOR gates: $F(w,x,y,z) = \sum(1, 2, 13, 14)$





$F = \Sigma(1, 2, 13, 14)$

$F' = w'x + wx' + y'z' + yz = [(w + x')(w' + x)(y + z)(y' + z')]'$

$F = (w + x')' + (w' + x)' + (y + z)' + (y' + z')$