CS1101: Lecture 20

The Digital Logic Level: The Sum of Products Notation

Dr. Barry O'Sullivan
b.osullivan@cs.ucc.ie

Course Homepage
http://www.cs.ucc.ie/~osullb/cs1101

Lecture Outline

- Boolean Algebra
- Describing Boolean Functions
- The Sum of Products Notation
- The Majority Function
- Decribing the Majority Function
- Implementation of Boolean Functions
- Using only NAND and NOR Gates
- Circuit Equivalence
- **Reading**: Tanenbaum, Chapter 3, Section 1

From Three to Five Simple Gates

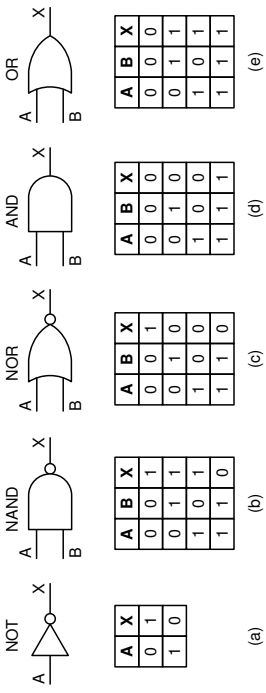


Figure 3-2: The symbols and functional behaviour for the five basic gates.

Boolean Algebra

- To describe the circuits that can be built by combining gates, a new type of algebra is needed, one in which variables and functions can take on only the values 0 and 1.
- Such an algebra is called a Boolean algebra.
- George Boole (1815-1864).
- A Boolean function has one or more input variables and yields a result that depends only on the values of these variables.
- A simple function, f, can be defined by saying that f(A) is 1 if A is 0 and f(A) is 0 if A is 1.
- This function is the NOT function

Describing Boolean Functions

- Boolean functions of n variables have only 2^n possible combinations of input values
- Thus, a function can be completely described by giving a table with 2^n rows, each row telling the value of the function for a different combination of input values – a **truth table**.
- A function can be also completely described by using the 2^n-bit binary number obtained by reading the result column of the truth table vertically.
- Thus NAND is 1110, NOR is 1000, AND is 0001, and OR is 0111.
- Only 16 Boolean functions of two variables exist

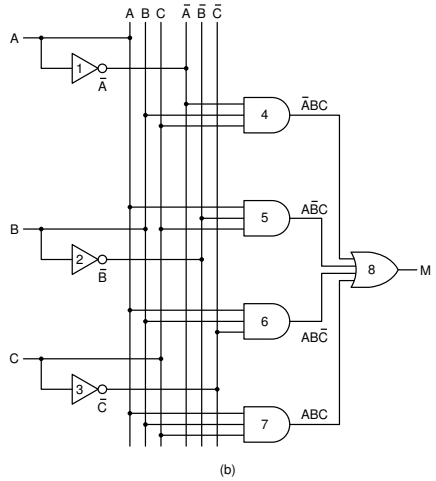
The Sum of Products Notation

- We will consider an example.
- We will consider the truth table for a Boolean function of three variables: $M = f(A, B, C)$.
- In particular, we will consider **the majority logic function**, that is, it is 0 if a majority of its inputs are 0 and 1 if a majority of its inputs are 1.
- Although any Boolean function can be fully specified by giving its truth table, as the number of variables increases, this notation becomes increasingly cumbersome.
- Instead, another notation is frequently used.

The Majority Function

A	B	C	M
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(a)



(b)

Figure3-3. (a) The truth table for the majority function of three variables. (b) A circuit for (a).

An Alternative Notation

- Note that any Boolean function can be specified by telling which combinations of input variables give an output value of 1.
- By convention, we will place a bar over an input variable to indicate that its value is inverted – the absence of a bar means that it is not inverted.
- We will use implied multiplication or a dot to mean the Boolean AND function and + to mean the Boolean OR function.
- Thus, for example, $A\overline{B}C$ takes the value 1 only when A = 1 and B = 0 and C = 1.
- Also, $A\overline{B} + B\overline{C}$ is 1 only when (A = 1 and B = 0) or (B = 1 and C = 0).

The Majority Function

- Thus, the majority function we considered can be defined as follows:

$$M = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

- A function of n variables can thus be described by giving a "sum" of at most 2^n-variable "product" terms.

- This formulation is especially important, as we will see shortly, because it leads directly to an implementation of the function using standard gates.

Implementation of Boolean Functions

1. Write down the truth table for the function.
2. Provide inverters to generate the complement of each input.
3. Draw an AND gate for each term with a I in the result column.
4. Wire the AND gates to the appropriate inputs.
5. Feed the output of all the AND gates into an OR gate.

Using only NAND and NOR Gates

- It is often convenient to implement circuits using only a single type of gate.

- Both NAND and NOR gates are said to be **complete**, because any Boolean function can be computed using either of them.

- One way to implement a Boolean function using only NAND or only NOR gates is:
  1. Follow the procedure given above for constructing it with NOT, AND, and OR.
  2. Then replace the multi-input gates with equivalent circuits using two-input gates.
  3. Finally, the NOT, AND, and OR gates are replaced by the following circuits.

Using only NAND and NOR Gates

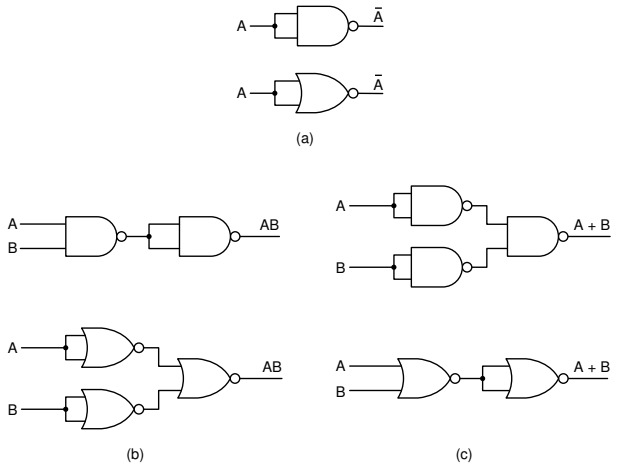


Figure 3-4: Construction of (a) NOT, (b) AND, and (c) OR gates using only NAND gates or only NOR gates.