CS 10 – Homework 3

Hansen Wu

2014.02.03

Foothill College

B.1, B.2, B.4, B.6, B.37

B.2, B2, B2, B2, B10

**B.1**

In addition to the basic laws we discussed in this section, there are two important theorems, called DeMorgan’s theorems:

A+B = A\*B and A\*B = A+B

Prove this using a truth table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | A | B | A+B | A\*B | A\*B | A+B |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

The results for A+B is equivalent to A\*B, and the results for A\*B are equivalent to A+B. From this outcome, it can be inferred that they are equal.

**B.2**

Prove that the two equations for E in the example starting on page B-7 are equivalent by using DeMorgan’s theorems and the axioms shown on page B-7.

**Unsolved**. I tried using the logic notation in Java, but got lost.

E is defined as “true if exactly two inputs are true”. What the first equation is essentially testing is: ((A && B) or (A && C) or (B && C)) && !(A && B && C)

Try applying the theorems to it:

( (A && B) || (A && C) || (B && C) ) && !(A && B && C)

( (A && B) || (A && C) || (B && C) ) && !(A && B && C)

!!( ( !!(A && B) || !!(A && C) || !!(B && C) ) && !(A && B && C) )

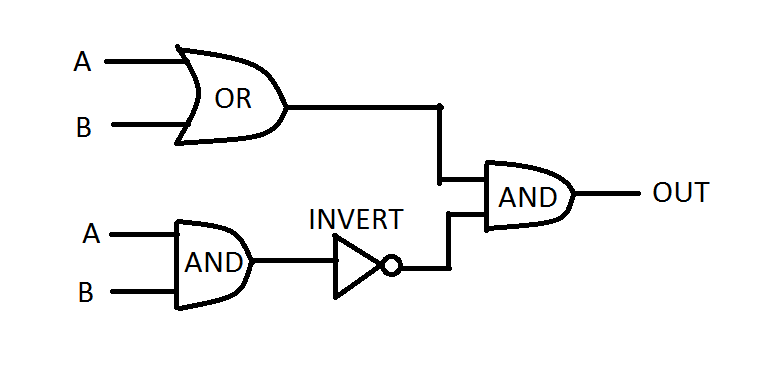
( !( !(!A || !B) || !(!A || !C) || !(!B || !C) ) || (A && B && C) )

( !( (A && B) || (A && C) || (B && C) ) || (A && B && C) )

( !( (A && B) || (A && C) || (B && C) ) || (A && B && C) )

**B.4**One logic function that is used for a variety of purposes (including within adders and to compute parity) is exclusive OR. The output of a two-input exclusive OR function is true only if exactly one of the inputs is true. Show the truth table for a two-input exclusive OR function and implement this function using AND gates, OR gates, and inverters.

|  |  |  |
| --- | --- | --- |
| A | B | XOR |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



**B.6**

Prove that the NAND gate is universal by showing how to build the AND, OR, and NOT functions using a two-input NAND gate.