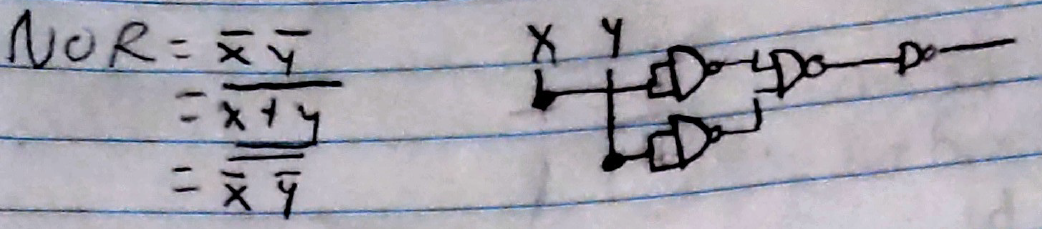
CPSC3300 – Computer Systems Organization  
Homework #2 – Boolean Algebra and Adders

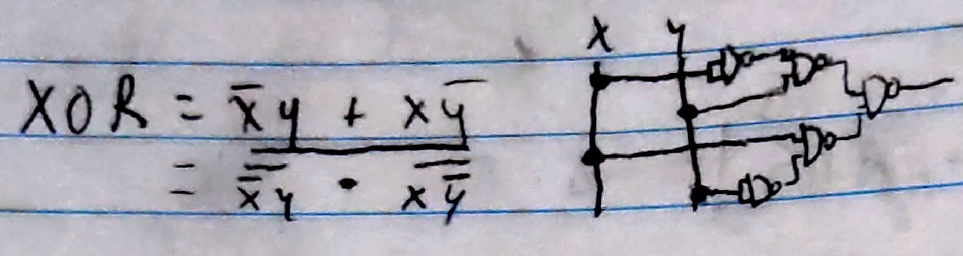
Due: 11:59PM Monday, February 10

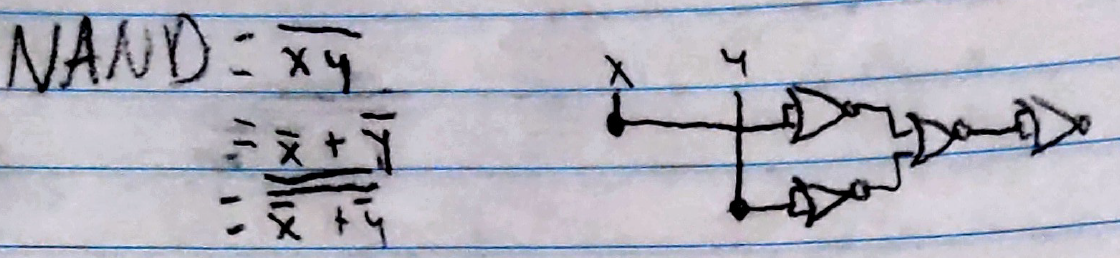
Submit to Canvas

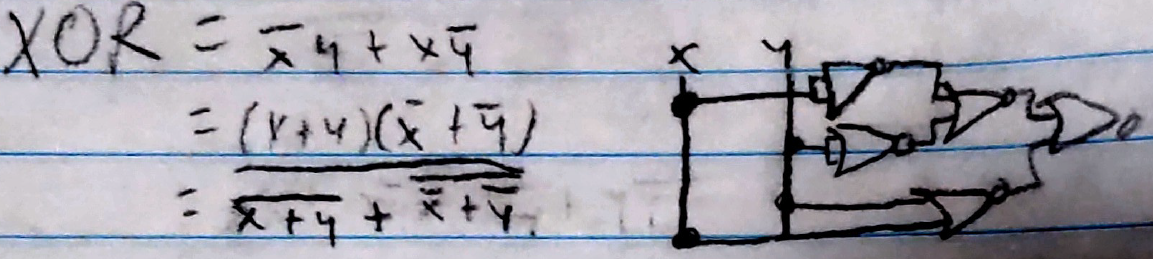
Total 100pts

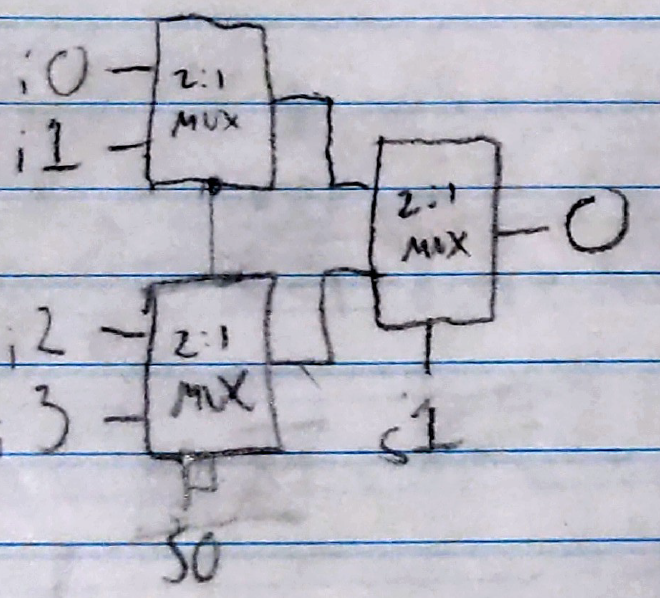
1. [20pts] Logical completeness
   1. Show that you can use only two-input NAND gates to implement each of the following two-input logic functions, and draw the used NAND gates and wiring.
      1. NOR function



* + 1. XOR function
  1. Show that you can use only two-input NOR gates to implement each of the following two-input logic functions, and draw the used NOR gates and wiring.
     1. NAND function



* + 1. XOR function

1. [10pts] Show how to use 2-1 Muxes to build a 4-1 Mux. Draw the used 2-1 Muxes and the wiring, and mark the 4 inputs and 1 output for the resulting 4-1 Mux.
2. [10pts] Demonstrate by means of truth tables whether the following identities are valid or not:

(O1) (O2) Not Valid

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **A** | **B** | **C** | **O1** | **O2** |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 |

(O1) (O2) Valid

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **A** | **B** | **C** | **O1** | **O2** |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

1. [20pts] Prove the identity of each of the following Boolean equations, using algebraic manipulation:

   2. 1
2. [20pts] For the Boolean function O1 and O2, as given in the following truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **x** | **y** | **z** | **O1** | **O2** |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

* 1. List the minterms for a three-variable function with variables x, y, and z.

|  |
| --- |
| x’y’z’ |
| x’y’z |
| x’yz’ |
| x’yz |
| xy’z’ |
| xy’z |
| xyz’ |
| xyz |

* 1. Express O1 and O2 in sum-of-product algebraic form.
     1. O1: x’y’z’ + xy’z + xy’z’ + xy’z + xyz
     2. O2: x’y’z + xy’z’ + xyz’ + xyz’

1. [20pts] In class, we learned the implementation for a 4-bit carry lookahead adder. We can use the same idea and extend to build a 16-bit carry lookahead adder. Denote this implementation as a one-level carry lookahead adder.

In the textbook, Figure 8.6.3 shows a two-level implementation of a 16-bit carry lookahead adder. This adder uses 4-bit carry lookahead adders at the lower level, and uses a carry lookahead unit at the higher level.

Compare these two implementations and provide your explanation why the two-level implementation could be preferred.

* Due to the fact that each bit needs to take from every lower bit, the one level adder requires way more logic fates than the two-level, and is therefore less practical.