Hui Shi Li

Lab 1

Section 6

January 20, 2015

The logic design that came from the truth table was found by using the sum of products. Looking at the truth table, we find all the outputs of 1. Then we flip all the bits of x, y, z so that when "and" together, it produces a 0 everywhere else. After we take each product and add them together, creating a sum of products equation.

In part B, there are 48 transistors in the original design. Using DeMorgan's Law, the design was simplified to 38 transistors. Instead of using "or", it simplified to only using "not" and "nand" gates. First, by inverting the entire equation, you get a nand of x, y, z "or" together. Furthermore, inverting the inner equation, you get another nand.

"and" and "or" gates have more transistors than "nand" or "nor" because "nand" and "nor" can take the same inputs but can invert the input using the same gate. To get the final circuit in part C, I factored out like terms to get, !x!y(!z+z) + yz(!x+x). Using Boolean algebra, specifically the complementation law, which is defined to be x + !x = 1. Then simplifying the equation to !(xy) + yz. In the final circuit, there are 22 transistors.

My guess at how the number generator works is that there is a complex algorithm that chooses a number between 0 and 1 such as .346. Then it determines if that number is closer to 0 or 1, like rounding up or down, and outputs whichever one is closer.