CMPE 12 Lab Report # 1

John Allard
TUTOR
Lab Section #2

October 6th, 2014

1 Overview

This lab served as an introduction to circuit design from truth tables using logic gates. All circuits were implemented using the free MultiMedia Logic (MML) program, which allows the user to drag, drop, and connect different circuit components. After the circuit is connected it can be simulated to help check for design errors. This lab consisted of 4 section, with each section varying in difficulty and length.

2 Part A : Using MultiMedia Logic

2.1 Procedure

This section was a combined introduction to MML and DeMorgan's laws. To start we were instructed to navigate to an MML tutorial on YouTube[1] and build the circuit that is shown in the tutorial. After this, we were instructed to show our understanding of MML by implementing DeMorgans laws $A'B' = (A + B)'$. To do this, we needed to build a circuit for each side of the DeMorgans equation and show their equivilence both in practice and by truth table.

2.2 Results

To start, we completed an extremely simple circuit on the MML software. This circuit is about as simple as one can get, it is just a binary switch connected to an LED. For the first circuit, the LED is connected directly to the switch. In this case, if the switch is high, the LED is on, if the switch is low, the LED is off. Next we simply inserted an inverter between the switch and the LED. These two circuits are shown below :

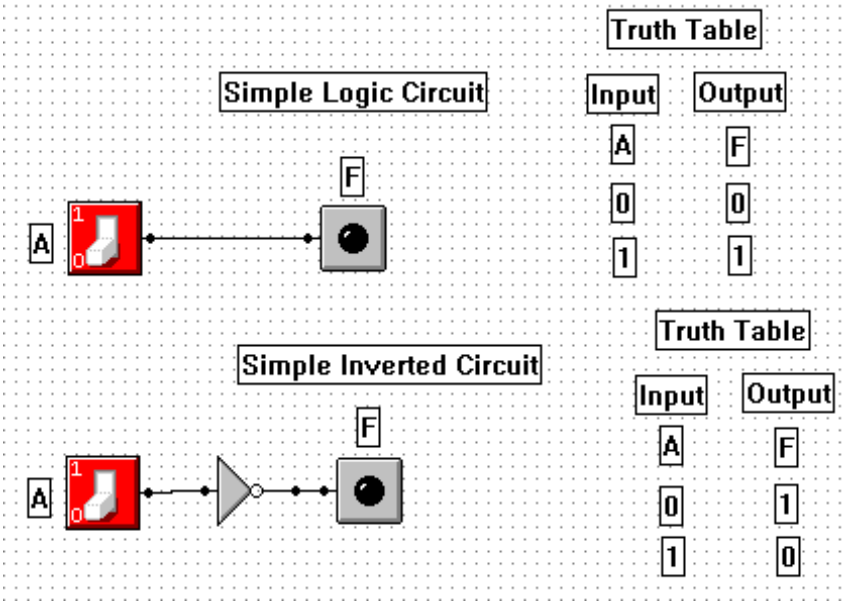

Figure 1: Circuit for Part A1, MML Tutorial.

Our next goal was to create two new circuits which demonstrait DeMorgans law, specifically the fact that $A'B' = (A + B)'$. This involved the use of two switch, labled A and B. These switches will be used and the inputs for two circuits, the first one representing $A'B'$ and the second one representing $(A + B)'$. After these circuits are complete, the simulation can be run and it can be seen that the LED for each of the two circuits are in an identical state for the span

[1] `http://www.youtube.com/watch?v=hJq2gECXYWc&noredirect=1`

of the message space, which happens to only be $\{00, 01, 10, 11\}$ for this simple example. The first circuit, $A'B'$, is created by passing the A and B inputs through inverters before combining them into an AND gate. The output of the AND gate is run directly to an LED. The second circuit is made by running the A and B inputs directly into a NOR gate, then to an LED. The circuit and truth tables can be seen below.

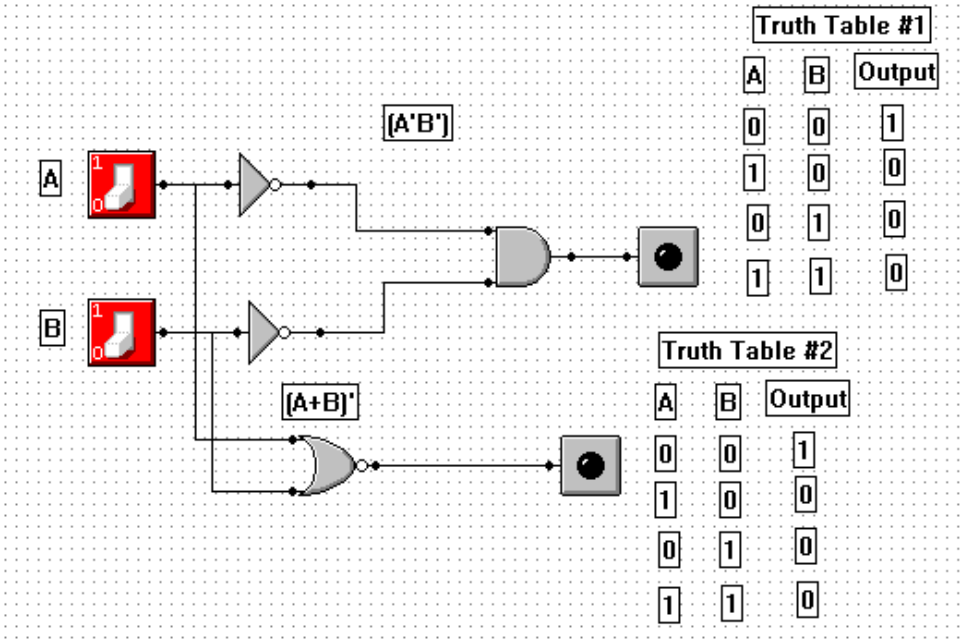

Figure 2: Circuits for Part A2, Showing DeMorgans Law.

3 Part B : Implementing Functions as Sums of Products

IN[2]	IN[1]	IN[0]	Output
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Figure 3: Truth Table for Part B

3.1 Procedure

This part of the lab consisted of two sections, B1 and B2. The goal of both sections was the same, implement the truth table in figure 3 (above), the differences between the two sections are the restrictions on the gates that can be used to implement the truth table.

In the circuits that we are building, the following names will be used for the inputs listed in figure 3 : IN[2] $\rightarrow A$, IN[1] $\rightarrow B$, IN[0] $\rightarrow C$ (reading left to right). The output of the

circuit will be a single LED with a basic 1 → HIGH, 0 → LOW relationship. Part B1 asked us to construct a circuit to represent the truth table in figure 3 using AND, OR, or Inverter gates. Part B2 asked us to reimplement the same circuit using only NAND gates. This requires translation of the AND, OR, and Inversion gates into nested layers of NAND gates. Finally, we were supposed to run the circuits in both sections side by side to ensure their equivalence.

3.2 Results

Before any circuitry could be properly constructed, the truth table had to be broken down into a boolean equation using the sum of products rule. Working from left to right in the truth table in figure 3, with IN[2] → A, IN[1] → B, IN[0] → C, we arrive at the following equation.

$$\text{A'BC'} + \text{A'BC} + \text{AB'C} + \text{ABC'} \tag{1}$$

Part B1 - AND, OR, & NOT Gates

Part B1 involved directly translating equation 1 into a circuit using MML. Because we were allowed to use AND, OR, and NOT gates, the circuit reads all most exactly like equation 1. The circuit for part B1 is shown in figure 4. The circuit is best read from left to right. It starts

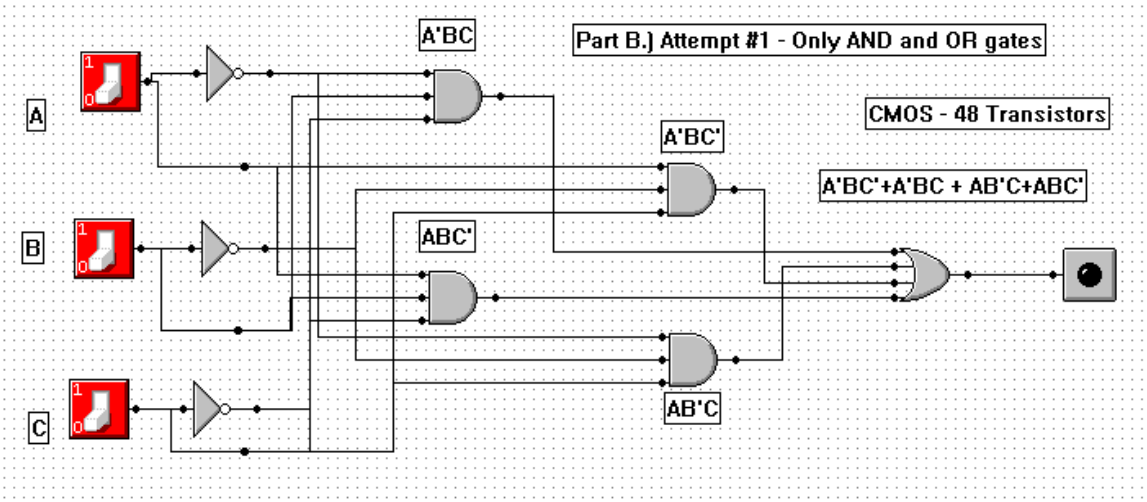

Figure 4: Circuit for Part B1

off with the three inputs, A, B, and C. These three switches can be toggled to create a total of $2^3 = 8$ possible combinations. Right of these switches is a layer of four 3-input AND gates. These AND gates represent the individual products in equation 1. The output from each of these four AND gates are routed to a single 4-input OR gate. This gate represents the sum of the individual products shown in equation 1.

Part B2 - NAND Gates