Rodrigo Becerril Ferreyra
CECS 361 Section 01
Lab 3
18 October 2020

1 Introduction

The purpose of this lab is to practice the verification method of equivalence checking. More specifically, we were tasked with creating an equivalent circuit based on one which was obfuscated (i.e. a "black box" circuit). Then, to verify that the two circuits are indeed equivalent, their respective outputs are XORd together—if this XOR is unsatisfiable (i.e. if it never results in a `1` when given all possible inputs), then the circuits are indeed equivalent. This circuit was later applied to our Nexys A7 100-T FPGA board.

2 Process

In this lab, we received a "black box" circuit, and were told to create an equivalent circuit based on its output. I determined that the output is combinatorial, so I employed the method of creating a truth table, and simplifying it into a Boolean function using a Karnaugh map.

To produce the following output, I created a `for` loop that goes through all $2^5 = 32$ different combinations of inputs.

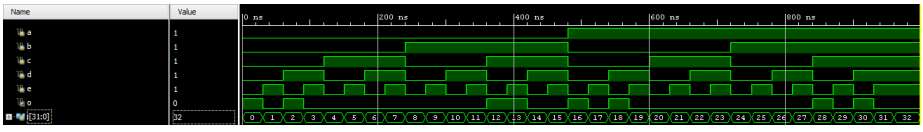

Figure 1: The output of the obfuscated module.

Below are the results compiled from the waveform above.

A	B	C	D	E	out
0	0	0	0	0	1
0	0	0	1	0	1
0	1	1	0	0	1
0	1	1	0	1	1
1	0	0	0	0	1
1	0	0	1	0	1
1	1	1	0	0	1
1	1	1	1	0	1
Otherwise					0

Table 1: Truth Table for obfuscated circuit.

In short, there are eight different combinations of inputs that result in a high output.

The following is a Karnaugh map simplification of the above truth table. The different colors represent groupings of the minterms: the four red bits are grouped together, the two green bits are grouped together, and so on. Using the rules of Karnaugh maps, this minimizes to the following Boolean function:

$$f(A, B, C, D, E) = \bar{B}\bar{C}\bar{E} + ABC\bar{E} + \bar{A}BC\bar{D} \tag{1}$$

ABC\DE	00	01	11	10
000	1			1
001				
011	1	1		
010				
110				
111	1			1
101				
100	1			1
f = B'C'E' + ABCE' + A'BCD'				

Figure 2: Karnaugh map of truth table.

Making an equivalent circuit to reflect the given circuit is as simple as making a module with Equation 1 inside it. The following is a waveform with both the `original` output (as shown in Figure 1) and the `equivalent` output from the equivalent circuit. Note how their waveforms are the same.

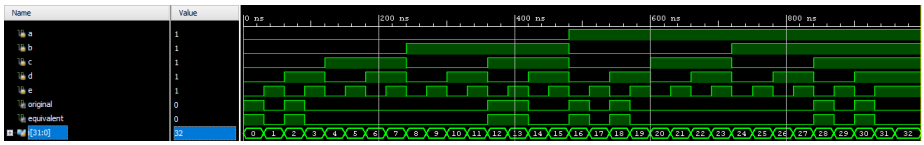

Figure 3: Waveform for both original and equivalent citcuits.

Lastly, I made a top-level module that drives the two other modules with the same inputs, takes their outputs, and XORs them together. Below is the waveform output of this module. Note how the `eq` signal never goes high; it is not satisfiable, and therefore the two circuits are equivalent.

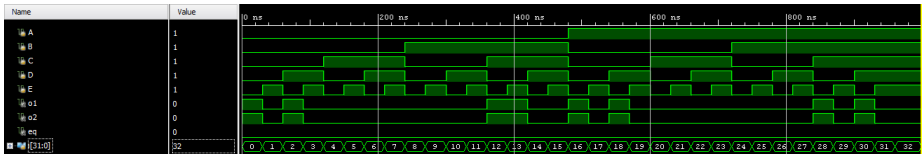

Figure 4: Waveform for top-level function.

Lastly, this was loaded onto the FPGA and tested for compatibility with no flaws or issues.

3 Media

Below are pictures of the board when the original and equivalent circuits are on and off.

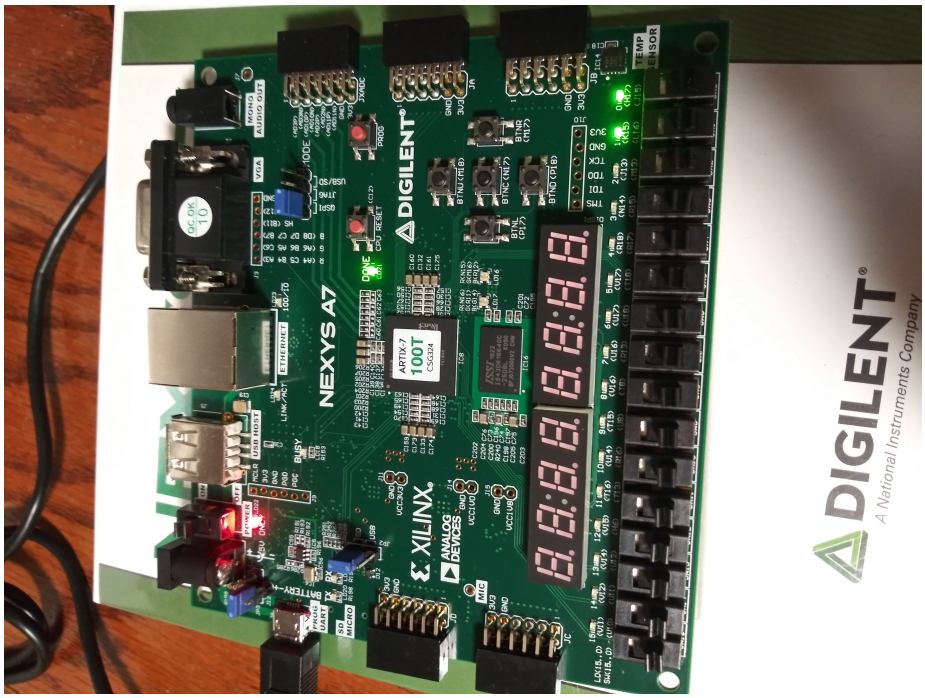

Figure 5: {a, b, c, d, e} = 0

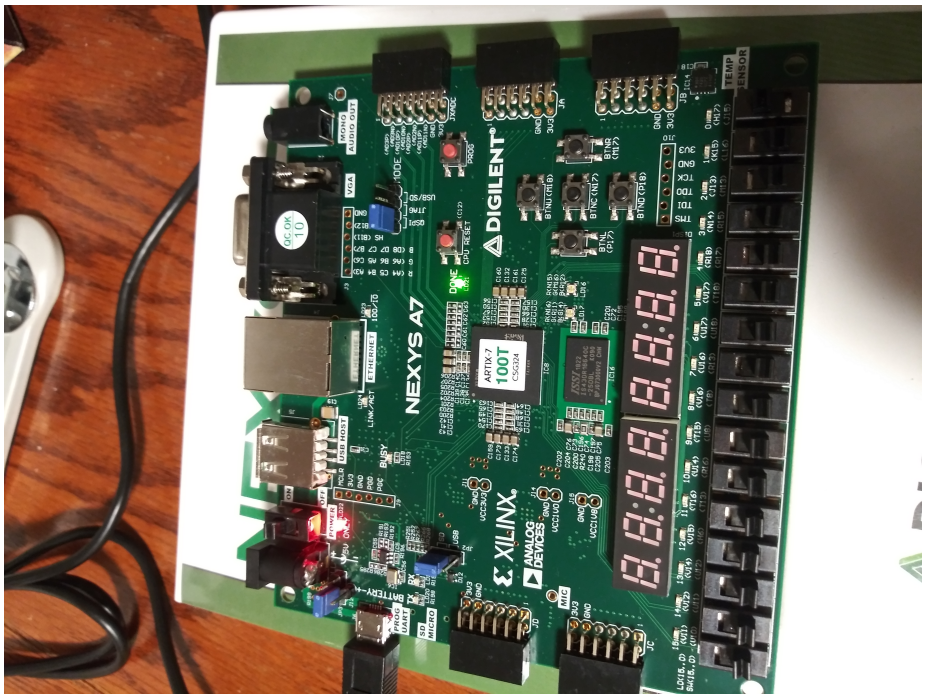

Figure 6: {a, b, c, d, e} = 1