CS M51A and EE M16 Summer 2016 Section 1

Logic Design of Digital Systems

Dr. Yutao He

Project #1 - Orientation of Verilog and Vivado

Due: Sunday, July 17^{th} 2016

Name: Liu (Last) Jerry (First)

Student ID: 404474229

Discussion: 1A

Date: 07/16/2016

Result	
Correctness	
Creativity	
Report	
Total Score	

Project 1 Report, Title Page Above

Jerry Liu, 404474229

July 16, 2016

2 Project Requirement

In this project, we will use the Xilinx Vivado software and Verilog to implement the circuit that is specified in Figure 1.
Inputs: We have three inputs: x_2, x_1, x_0, with $x_i \in \{0, 1\}$.
Outputs: We have one output: z, with $z \in \{0, 1\}$.
Function: The canonical form of the output function is

$$z(x_2, x_1, x_0) = \sum m(1, 2, 6) = x_2' x_1' x_0 + x_2' x_1 x_0' + x_2 x_1 x_0' = x_2' x_1' x_0 + x_1 x_0'$$

3 The Function of The Circuit

In this implementation, we use the sum of minterms. The canonical switching expression is:

$$z(x_2, x_1, x_0) = \sum m(1, 2, 6) = x_2' x_1' x_0 + x_2' x_1 x_0' + x_2 x_1 x_0'$$

But we can simplify the function to be:

$$z(x_2, x_1, x_0) = x_2' x_1' x_0 + (x_2' + x_2) x_1 x_0' = x_2' x_1' x_0 + x_1 x_0'$$

Let prod1 $= x_2' x_1' x_0$, prod2 $= x_1 x_0'$. The corresponding diagram is shown in Figure 1:

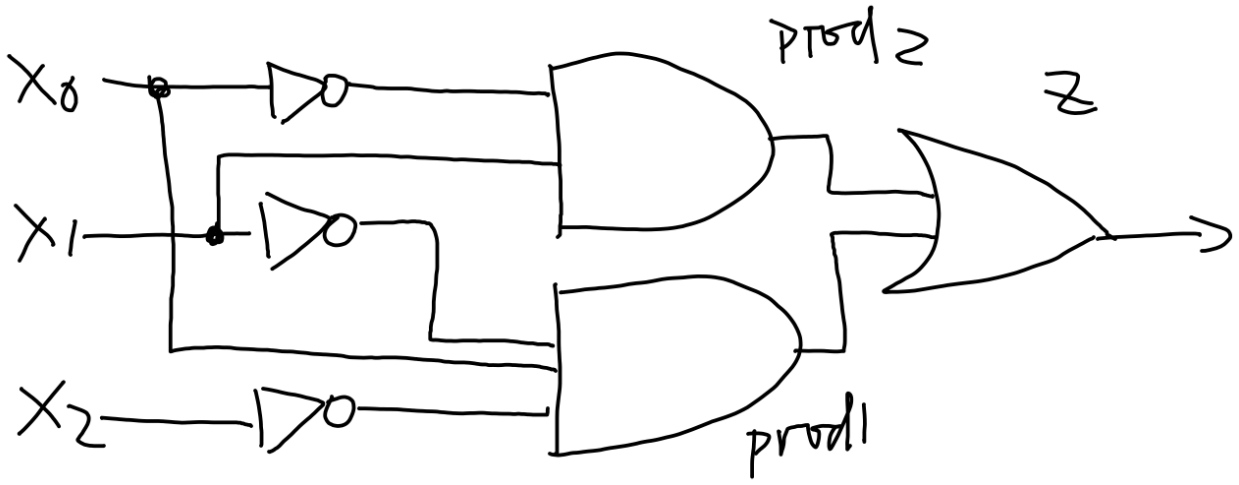

Figure 1: Schematic Diagram

4 Verilog File

4.1 Circuit Implementation

csm51a_proj1.v

```
module csm51a_proj1(
    input x0,
    input x1,
    input x2,
    output z
    );
    //  Outputs from NOT and AND gates
    wire x0_not, x1_not, x2_not, prod1, prod2;

    //  NOT inputs
    not n1(x0_not, x0);
    not n2(x1_not, x1);
    not n3(x2_not, x2);

    //  Calculate Products

    // prod1 = x2'x1'x0
    and a1(prod1, x2_not, x1_not, x0);

    // prod2 = x1x0'
    and a2(prod2, x1, x0_not);

    //  Calculate Sum of Products

    // z = prod1 + prod2
    or o1(z, prod1, prod2);

endmodule
```

4.2 Testbench

csm51a_proj1 tb.v

```
module csm51a_proj1_tb(
    output z
    );
    //  4-bit integer to avoid overflow from 3-bit integers
    reg [3:0] i;
    //  Input variables
    reg x0, x1, x2;

    //  Test out implementation
    csm51a_proj1 p1(.z(z), .x0(x0), .x1(x1), .x2(x2));
```

```
    initial
    begin
        for (i = 0; i < 8; i = i + 1)
        begin
            // assign each bit separately
            {x2, x1, x0} = i;
            // delay
            #10;
        end
        $finish;
    end
endmodule
```

5 Simulation Result

Here we want to construct and test all possible cases in the truth table: Table 1.

x_2	x_1	x_0	z	m_i
0	0	0	0	m_0
0	0	1	1	m_1
0	1	0	1	m_2
0	1	1	0	m_3
1	0	0	0	m_4
1	0	1	0	m_5
1	1	0	1	m_6
1	1	1	0	m_7

Table 1: Truth Table of the Function

The results are shown clearly in the Figure 2:

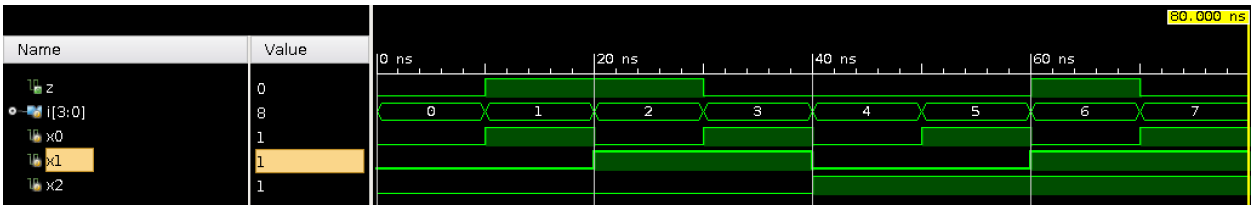

Figure 2: Screenshot of the Simulation Result

As we can see, the output $z = 1$ if and only if the inputs $(x_2, x_1, x_0) = (0, 0, 1), (0, 1, 0), (1, 1, 0)$ (Corresponding to $i = 1, 2, 6$). This is exactly what we get from the truth table: Table 1.

Note that green regions correspond to the time for a specific variable to have value 1, and for each variable, its value variations are represented on the same line as the variable.

In Figure 3, we can get a more detailed view of variable's values when $z = 1$.

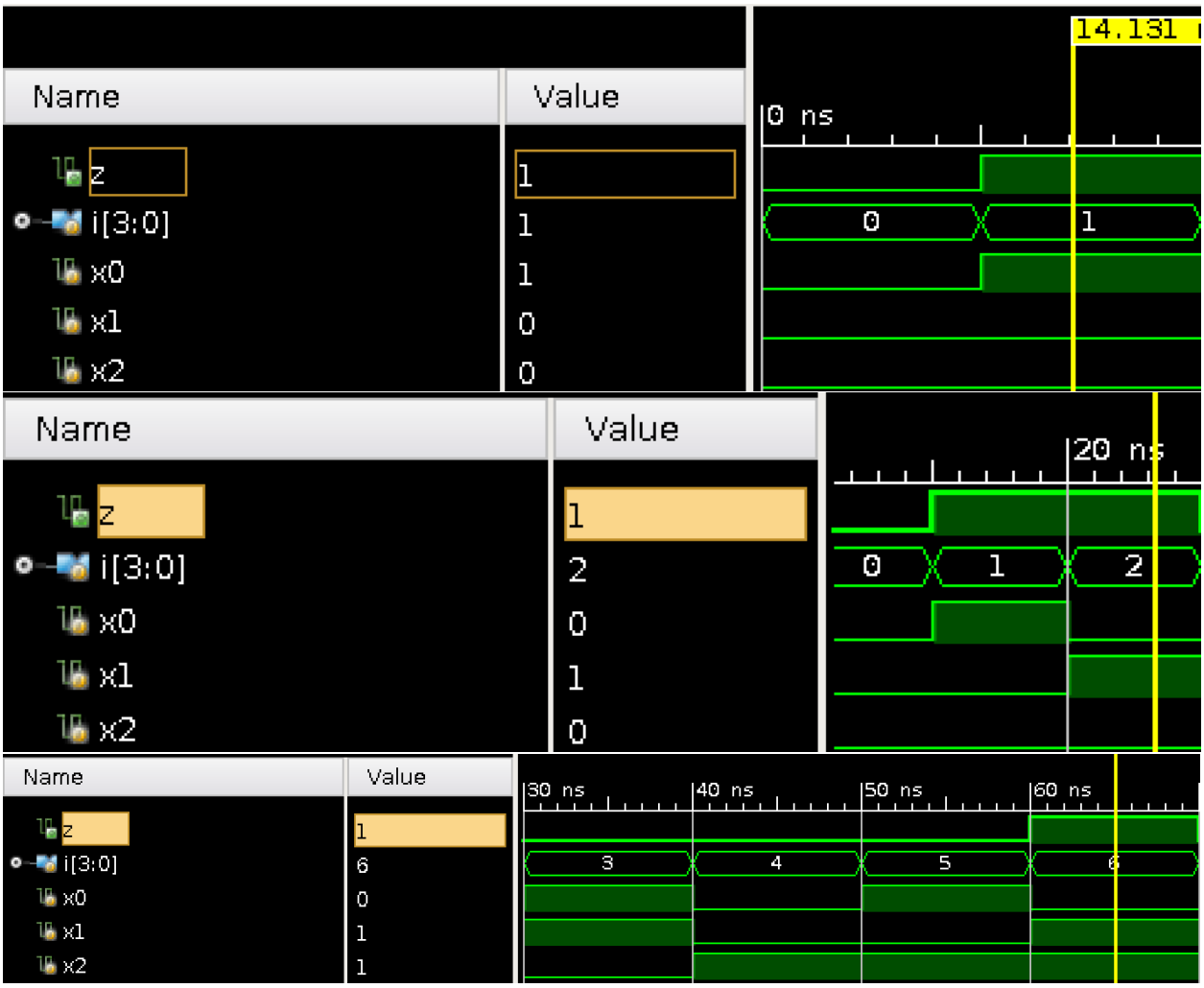

Figure 3: Screenshot of the Simulation Result for Minterms m_1, m_2, m_6

6 The Summary

6.1 Issues Encountered

I tried to implement a *for loop* to run the TestBench, but I had a hard time figuring that out. I asked my TA about it and I was then able to figure out that I need to add a semi-colin in the delay. This is different inside for loops.