

CSE 369 Lab 1-2

An Introduction to SystemVerilog and Digital Components

Isaac Wu
2360957
October 16, 2024

1. SystemVerilog Design and Simulation

A simple, accurate explanation of what the mux4_1 circuit does. Note: We do NOT want a written description of the circuit gates, we want a description of what it actually does – if you're not sure, see how we described how the mux2_1 circuit works in Section 2 of the Quartus Tutorial.

mux4_1 has 4 inputs: i00, i01, i10, i11, along with 2 select inputs: sel0 and sel1. Based on the selectors, the mux4_1 circuit will match the corresponding input. sel0 represents the rightmost bit of the input and sel1 represents the left. For example, if sel0=0 and sel1=1, then mux4_1=i10.

2. Logic Investigation

1. Which logical value (0 = FALSE = GND, 1 = TRUE = VDD) turns the red LEDs on?

1/VDD turns the red LEDs on.

2. Which position (up or down) of the slider switch outputs a TRUE?

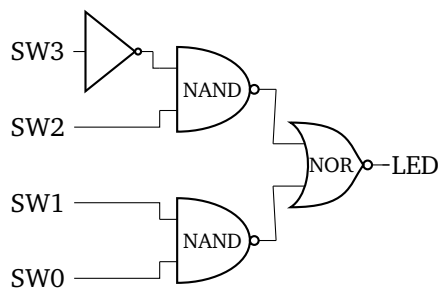
The up position of the slider outputs a TRUE.

3. Which position of the push button (pressed or unpressed) outputs a TRUE?

The unpressed position of the push button outputs a TRUE.

4. Digit Recognizer (Design)

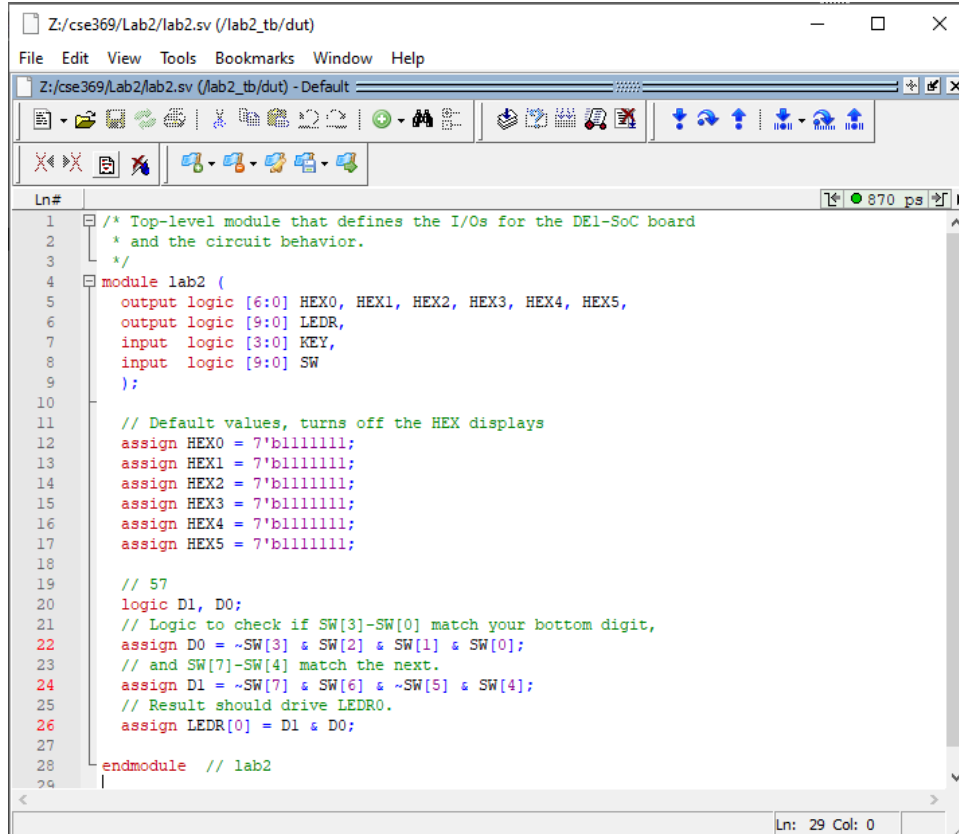
1-digit recognizer circuit:



5. Multi-Digit Recognizer (Implementation)

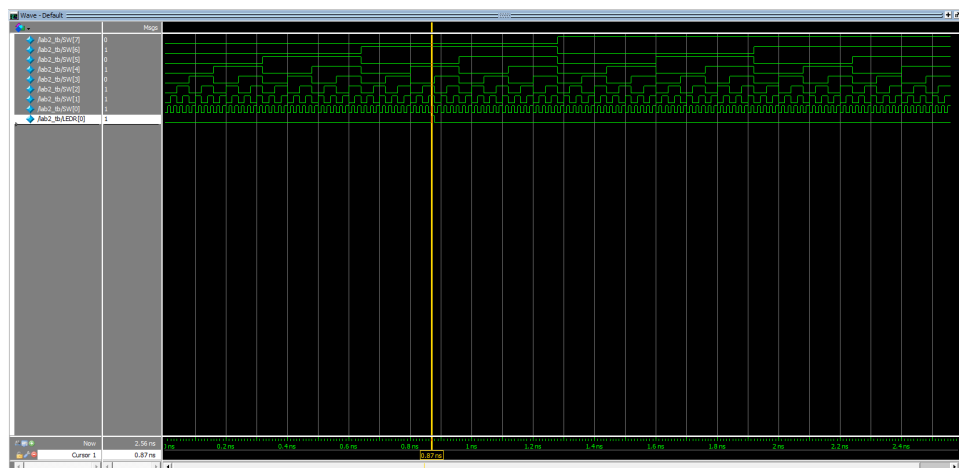
A screenshot of the ModelSim simulation for your 2-digit recognizer design (always with explanation!).

Here is the code for my multi-digit recognizer. I defined two logic intermediaries to keep track if my input matches each digit. Then, I combined them and assigned them to the LED output.



```
1  /* Top-level module that defines the I/Os for the DE1-SoC board
2  * and the circuit behavior.
3  */
4  module lab2 (
5      output logic [6:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5,
6      output logic [9:0] LEDR,
7      input logic [3:0] KEY,
8      input logic [9:0] SW
9  );
10
11      // Default values, turns off the HEX displays
12      assign HEX0 = 7'b1111111;
13      assign HEX1 = 7'b1111111;
14      assign HEX2 = 7'b1111111;
15      assign HEX3 = 7'b1111111;
16      assign HEX4 = 7'b1111111;
17      assign HEX5 = 7'b1111111;
18
19      // 57
20      logic D1, D0;
21      // Logic to check if SW[3]-SW[0] match your bottom digit,
22      assign D0 = ~SW[3] & SW[2] & SW[1] & SW[0];
23      // and SW[7]-SW[4] match the next.
24      assign D1 = ~SW[7] & SW[6] & ~SW[5] & SW[4];
25      // Result should drive LEDR0.
26      assign LEDR[0] = D1 & D0;
27
28  endmodule // lab2
```

This is what my wave diagram looks like. The top 8 waves show the 8 switches alternating to test all 256 combinations. The bottom wave is the LED output, which is only true in one combination: 01010111.



6. Misc.

How many hours (estimated) it took to complete this lab in total, including reading, planning, designing, coding, debugging, and testing.

It took around 5-6 hours to complete this lab in its entirety.