Lab Report Title Page

Christopher Padilla

Lab 1

ECEN 449-504

Purpose/Introduction

- This lab is intended for students to become more familiar with the way Verilog works and how we can transform implementations into hardware using a field-programable gate array. It is meant as an introduction/refresher to using Vivado, as well as the different functions in the FPGA including buttons, switches and the clock.

Procedure

- This lab was divided into three parts. The first part of the manual demonstrated how to map inputs/outputs from Verilog to the hardware. This was self-explanatory in the lab manual and very little brainwork went into this part.
- The second part served as an example of a possible scenario where we may use the FPGA's clock function. Since we had to count up and down every second and the FPGA's clock is 125MHz, I divided the clock by 125,000,000 so that it only had a frequency of 1 Hz.
- The final part with the jackpot machine was a second example of how the clock could be used. The LEDs needed to blink one at a time in sequence from 1000 to 0001. I used a bunch of test conditions where it switched to the next pattern in the sequence depending on the current pattern, and if the corresponding switch was up, all the LEDs stayed on. I only divided the clock by 10,000,000, because the sequence of blinking lights to be fast but not too fast/slow. I reused the clock divider module that I created in part 1 but just changed the value by which I was dividing the clock.

Results

- The main part of this lab is creating a working clock divider module. An always@ block for the FPGA's main clock that is triggered at its positive edge is necessary, and every n number of clock cycles I set my new clock to be a 1, otherwise it would be a 0. Mapping the two (three if we include reset) buttons to whether they count up or down and using test conditions to determine which button was being pressed at that time was also necessary.
- I counted up using decimal; however, I mapped my LED and counter values using binary (this was to avoid doing a thousand if statements).
- Implementing the jackpot machine just involved me changing the values of the clock divider module (so it's faster than in part 1 but not too fast). Since there was only four different possible LED values, I just performed four corresponding if statements that depended on the previous value, while checking to see if the corresponding switch is high or low.

Conclusion

- All my results were as expected, and I was amazed when everything finally worked. After spending hours in the lab, I did get a nice refresher on how to implement hardware using Verilog HDL, mapping variables, and using trigger conditions. However, the transition between parts one and two of this lab went from 0 to 100 real quick, as part two was significantly tougher to implement than part one. I can only assumed the rest of the labs are going to be even harder, so I'm taking this as a learning experience

that not everything will be taught to me and I have to put a tremendous amount of effort into these labs.

Include Verilog Code

- My Verilog code is stapled to this report (Microsoft Word messes up my formatting).

Questions

- I mapped the user push-buttons that are wired on the ZYBO board by modifying the given .xdc file. The three buttons that I used (one for counting up, one for counting down, and one for reset) have PIN values of Y16, V16 and P16 respectively.
- Edge detection is used as a trigger condition. In this lab, we used the ZYBO board's clock function as a trigger condition. I used it to increment a counter up to 125,000,000 (10,000,000 for part two) and creating a new clock with a lower frequency. The positive edge of the new clock is what is used as a trigger condition to switch around the LED values (counting up, or alternating between them).