Microprocessor-Based Systems (E155)

Lab 2: Multiplexed Display

Learning Objectives

By the end of this lab you will have…

- Implemented a time-multiplexing scheme to drive two seven-segment displays with a single set of FPGA I/O pins.
- Built a simple transitor circuit to drive large currents.
- Practiced your ability to build Verilog systems in a modular way.

Requirements

Display two independent hexadecimal numbers on your dual seven-segment display. Use your DIP switch and four other input pins to provide the data for two hexadecimal numbers. You must use a single seven-segment decoder module to drive the cathodes for both digits on the display, which therefore must be wired for multiplexed operation. Also, display the sum of the numbers on five LEDs on your utility board.

Discussion

Time-multiplexing is a technique to share a common expensive hardware resource for several purposes at different times. For example, the multicycle processor in E85 multiplexed the memory for both instruction and data access and multiplexed the ALU for data processing instructions, branch calculations, and program counter increments. In this lab, you will time-multiplex your seven-segment decoder module to run both halves of a dual display.

A convenient way to control which half is active is to turn ON the common anode of only one display at a time. The anode requires substantial current, more than an FPGA output pin can drive. You can use a transistor to drive the large current. The lab has a stock of 2N3906 PNP transistors suitable for this purpose. Be sure to limit the base current. Choose a suitable switching speed. If you switch too slowly, your eye will notice the flicker. If you switch too fast for the electronics, the two digits will bleed together.

What to Turn In

When you are done, have your lab checked off by the instructor. You should thoroughly understand how it works and what would happen if any changes were made. Turn in your lab writeup including the following information:

- Schematics of the breadboarded circuit.

- Your Verilog code (and simulation results, if applicable)
- How did you choose your component values?
- How many hours did you spend on the lab? This will not count toward your grade.

Hints

Look at your RTL schematic in Quartus (*Tools -> Netlist Viewers –> RTL Viewer*). Understand why your code produces the hardware you see. Be sure your combinational logic doesn't have any registers. Be sure your logic has no latches or tristate buffers. The oscilloscope is handy for tracking down timing problems.