Lab Session #7

Overview

The object of this lab in digital design is to practice designing basic synchronous logic circuits using Verilog. Also, you should gain some familiarity with manipulating clock signals.

Preparation

- Review the logic design of the BCD to 7 segment display from class and lab last week.
- Review the modulo counter and shift register from class.

#1: “Easy” Task: Manipulating the UP2 Clock

The Altera UP2 board we’re using has a 25.175 MHz clock crystal built in to the board. (On the FLEX chip, it is attached to pin 91.) For the display purposes we’ll be using the clock for, we’d like a MUCH slower frequency – 1 Hz.

**Design a Verilog “Clock Divider”** – a module that creates a 1 Hz clock output given the 25.175 MHz clock as an input. The easiest way to accomplish this is to keep a count of “ticks” of the fast clock, and then, when it reaches a certain count, produce a toggle on the output. **Include a reset signal!** (it can be synchronous or asynchronous) (Hint: the Verilog ‘+’ operator is your friend!)

You can test your module pretty easily by compiling it, mapping pin 91 to the input, one of the pushbuttons to the reset, and one of the LED segments to the output. If you constructed the module correctly, the LED should turn on and off at the rate of 1 Hz (i.e. on -> off -> on takes 1 second).

#2: “Big” Task: Displaying a Date

A date is often represented by 3 sets of 2-digit numbers. For example, this lab’s due date could be written as:

04 – 08 – 09

(MM – DD – YY)

**Construct a Verilog module (or modules) to display a date on the two 7-segment displays.** Because you can only display two digits at a time, you’ll need to rotate the 3 sets of digits. For example, to display the date above, your circuit should first display “04” and then after a delay display “08” and then (after another pause) display “09”. Your design should repeat this pattern continuously.
You will want to store your date numbers in BCD format, and then translate them to 7-segment using
the conversion module you designed in lab 5. Use your clock division circuit to make the delays
something more reasonable that 25 MHz. Beyond that, there are LOTs of ways to implement this!
Spend some time considering your options. Here are a few suggestions:

- Store the 3 sets of numbers in a circular shift register (once something is shifted off the end, it is
  shifted back onto the beginning) and shift everything over every clock. Display the number that
  is sitting in a given position. Or...
- ... use a modulo counter to control a MUX that routes the various date numbers to the output.
  Or...
- ... come up with another idea.

If you think carefully, you may be able to keep the Verilog code short and sweet by writing the entire
rotation/counting setup in a behavioral manner (i.e. don’t instantiate separate shift register or counter
modules). This isn’t required, but might be worth considering.

Within your circuit, choose a date (your birthday, today’s date, whatever) and hard-code it into your
design. You could imagine having another module that keeps track of the current date (using the clock!) and passing it to this module, but we’ll skip implementing that for this lab.

Demonstrate your circuit and turn in your project files. (Zip up the entire project directory and drop it
in the W: dropbox.)

“Bonus” Task: Changing the Rate

If you’ve finished task #2 and can’t control your boredom, try designing a circuit that changes the rate at
which the date rotates based on inputs from the DIP switches. (i.e. 00 is 1 second, 01 is 2, 10 is 3, 11 is
4, etc.)