EECS 151/251A Homework 2

Due Monday, Feb 8th, 2021

For this HW Assignment

You will be asked to write several Verilog modules as part of this HW assignment. You will need to test your modules by running them through a simulator. As shown in discussion 2, a highly suggested simulator is `https://www.edaplayground.com` which is a free, online, Verilog simulator.

For **all problems** turn in:

1. Circuit Diagram (neatly drawn by hand or with a tool). When drawing there are some rules we'd like you to follow:
   - Draw solder dots at wire junctions
   - Label bus widths for N-bit wires
2. Verilog code, test bench, and test results.
3. A short (1-2 sentence) description of your code and how you arrived at your solution.

Warning: As per our EECS151 "no register inference policy", you will need to use the register library in `EECS151.v` when using registers in your Verilog. `EECS151.v` is located at `https://inst.eecs.berkeley.edu/~eecs151/sp21/files/lib/EECS151.v`. We will only accept solutions using this library!

You can import this library by adding `` `include "EECS151.v" `` to your Verilog code.

Examples of creating a testbench can be found in discussion 2 slides `https://inst.eecs.berkeley.edu/~eecs151/sp21/files/discussion2.pdf`.

Problem 1: Multiplexer

Design a 4-to-1 multiplexer using only continuous assignment (i.e. no `always` blocks). Provide an exhaustive test.

Problem 2: More Multiplexers!

Consider a 6-to-1 multiplexer, with a 6-bit wide data input `X`, 3-bit wide `select` input, and single bit output `y`. If `select` is 000, then `y=x[0]`, if `select=001`, then `y=x[1]`, etc. When `select=110` or `select=111`, the output is unspecified (i.e. "don't care").

(a) Using the definition of a 2-to-1 multiplexer from the class notes, define a hierarchical digital system to represent this 6-to-1 mux.

(b) Repeat (a), but use a `case` block with a flat (i.e. no instantiation) module.

You can submit one testbench that tests both designs together.

Problem 3: Decoder

For a 2-bit binary decoder, devise two behavioral Verilog descriptions without using the `case` statement (*hint: conditional statements and Boolean Logic*).

You can submit one testbench that tests both designs together.

Problem 4: Serializer

Many modern high-speed communications are sent over serial links. This means that the data buses from the processor must be serialized before going to the transmit block to be sent over the link. On slide 22 of Lecture 4, there is an example of a 4-bit parallel-to-serial converter (also known as a serializer).

(a) Adapt this design to be parametrizable to N-bits, but also modify it to shift in 0s instead of rotating the bits.

(b) Instantiate a 4-bit version of the serializer and load with the value `4'b0110`. Simulate for 10 cycles and take a screenshot of the waveform.

Note: when using registers in your Verilog, you will need to use the register library in `EECS151.v`.

Problem 5: Wrap-Around Counter

Design an N-bit parametrizable wrap-around counter with separate `up` and `down` control inputs. When both control inputs are 1, the `up` input takes priority, that is the counter will count up in such a situation. If both signals are 0, the counter stops counting. Test your counter for N=3. The testbench should cover the following situations:

- Count up from 0 and wrap around
- Count down from maximum and wrap around

- Count up with both control inputs 1 and wrap around
- Stop count at 0 with both control inputs at 0
- Count up halfway then stop the count for 2 clock cycles, then count down to 0

Problem 6: Accumulator

Consider the accumulator on Lecture 4 Slide 10. Design a parametrizable version for N-bits. Test exhaustively for N=4

Problem 7: 251A only — *Optional* Challenge Question for 151

Modify the accumulator in Problem 6 to have a parametrizable saturation value (both positive and negative 2's complement), such that when the accumulator reaches this value, it will no longer increment or decrement.