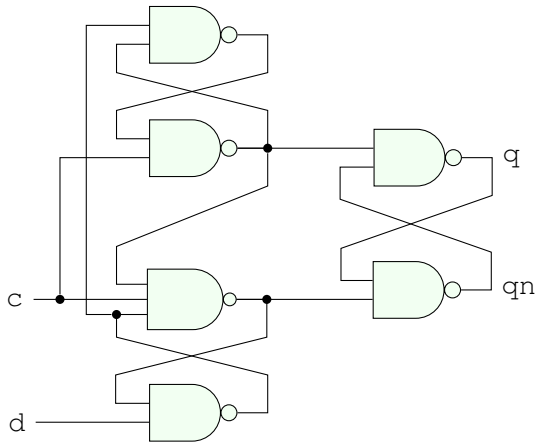
ECEN 2350: Digital Logic

Assignment #9

1. [5 points.] The file `lab9-q1.v` that you can download from Canvas, contains a testbench for the following circuit, which implements a positive edge-triggered D flip-flop.



   Complete `lab9-q1.v` by writing a module that describes the above circuit using gate-level Verilog primitives. Each of the six gates should have a unit delay. Recall that a NAND gate with unit delay, output `c`, and inputs `a`, `b` is written

```
nand #1 (c, a, b);
```

   (A primitive gate instance also has an optional instance name, not shown above.) Make sure your simulation results confirm that the circuit you described is a positive edge-triggered D flip-flop. For that, check when the outputs of the circuit change in response to changes in `c` and `d`. Submit your completed `lab9-q1.v` file.

2. [2 points.] For the above flip-flop, what is the delay (in simulation time units) from the rising edge of the clock to when the Q-output latches the data? Is it the same if the data is a 0 or 1?

   Provide your answer and explanation in a file `lab9-q2.txt`.

3. [8 points.] Make a **shift register** on the Boolean board that works as follows:

   The LEDs should display the contents of a 16-bit shift register, that shifts to the right every time push button 0 (e.g. `btn[0]`) is pressed. The shift register should shift in the value of the left-most switch (`sw[15]`).

   Note: if you want to use `btn[0]` as a clock (e.g. for use in an `always @(posedge btn[0])` statement), you will need to tell Xilinx to treat the button as a clock input. To do this, add the following code to your constraint (.xdc) file:

   ```
   # Add this to allow btn[0] to be used as a clock
   set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn[0]];
   ```

   If you don't add this, you may get an error stating `Poor placement for routing between an IO pin and BUFG` that prevents your design from synthesizing. Adding the above to your .xdc file should fix the issue.

   A video demonstrating the desired shift-register functionality can be found here: https://www.youtube.com/watch?v=ef7ocEqFAoM

   Turn in your Verilog file implementing the shift register as `lab9-q3.txt`.

4. [10 points.] Make a simple 16-bit adding calculator. Your calculator should have two 16-bit operands, A and B.

   When you press `btn[0]`, operand A should store the 16-bit number provided on the switches (`sw[15:0]`). When you press `btn[1]`, operand B should store the switch value.

   At all times, you should display the 16-bit sum $A + B$ in binary on the LEDs.

   A video demonstrating the desired functionality can be seen here (enable subtitles to see the A/B register values): https://youtu.be/kUoNIFW2XNg

   Turn in your Verilog file as `lab9-q4.txt`.