ECE 425 – VLSI Circuit Design  
Laboratory 9 – Sequential Design with Verilog  
Spring 2007

Introduction

In this lab you will first extend a simple Verilog description of a 4-bit counter and simulate it to verify that it operates properly. You will then use the Synopsys tools to synthesize logic that implements your modified counter and the db2mag script to generate a Magic layout of your circuit. Finally, to verify that the synthesized circuit operates properly you will extract this circuit and simulate it using IRSIM. Next, you code a Verilog description of a "universal shift register" from scratch and repeat the process.

For the first part of the Lab, we will start with the following Verilog description of a 4-bit counter with synchronous clear:

```verilog
module counter(clk, clr, Q, carry);
    input clk, clr;
    output [3:0] Q;
    output carry;
    reg [3:0] Q;
    assign carry = (Q == 4'b1111);
    always @(negedge clk)
        begin
            if (clr) Q <= 4'd0;
            else Q <= Q + 1;
        end
endmodule
```

Prelab

1. Modify the code shown above by adding an input called "updown" that controls the counting direction. Modify the always block so that when updown==1, the circuit counts "up", (i.e. 0, 1, 2, ..., 14, 15, 0...) and when updown==0, the counter counts "down" (i.e. 15, 13, 13, ..., 2, 1, 0, 15, ...).

2. Add a 1-bit "load" input and a 4-bit data input to the counter. Modify the always block so that when load==1, the Q output is assigned the data input, and when load==0, the counter performs its regular function as described in step 1.

3. Create a new Verilog module that will implement an 8-bit "Universal Shift Register". The inputs, outputs, and behavior of the module are summarized below. Note that when data is shifted left, the least significant bit should "shift in" the in_l input.
Similarly, when data is shifted right, the most significant bit should "shift in" the in\_r input.

<table>
<thead>
<tr>
<th>sel</th>
<th>New Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Old Q</td>
</tr>
<tr>
<td>1</td>
<td>data</td>
</tr>
<tr>
<td>2</td>
<td>Q shifted left</td>
</tr>
<tr>
<td>3</td>
<td>Q shifted right</td>
</tr>
</tbody>
</table>

In the Lab

1. Simulate your modified counter with Verilocker and verify that it operates properly.
2. FTP your Verilog file to a Linux system. Use the Synopsys tools to synthesize your counter following the steps in Lab 8 and generate a Magic layout for the counter circuit using the db2mag script.
3. Use Magic to view the layout of your counter circuit and record its size for your report.
4. Extract your circuit and simulate with IRSIM to show that it operates properly. Plot the counter output using the IRSIM analyzer to hand in with your report. To use IRSIM with a clock signal, use the "clock" command to set up the clock cycle with four steps low and four steps high.
   
   \texttt{irsim> clock clk 0 0 0 0 1 1 1 1} 
   
   Then use the "c"
   
   Simulate your circuit for 3-4 clock cycles with “clr” held high before attempting to verify the function of the circuit.
5. Repeat steps 1-4 for your shift register design.

Report

For your lab report hand in the following:

1. A short technical memorandum which describes what you did, what you learned, and any problems that you encountered.
2. Listings of your Verilog files.
3. Timing diagram plots for the simulation of your Verilog files.
4. Schematic diagram plots of your synthesized designs from design\_vision.
5. Plots of the layout of your synthesized designs, marked with the dimensions of the overall cell.
6. IRSIM plots showing the proper simulation of the extracted Magic layouts.