CMPE 415 - Homework #4

Spring 2015

Start assignment as soon as possible. Work individually, but you can ask your classmates for help when you get stuck, with consideration to the course collaboration policy (please read it in the course website). Please send me email if something isn't clear and I will update the assignment. Changes are logged at the bottom of this page.

Before getting started, you should go through the verilog notes located under Course Readings on the course home page.

Notes:

- [15% of points] Clearly state whether your design is fully functional, and state the failing sections if any exist.
- Make sure your design and code are easily readable and understandable (clear and well commented).
  - Up to 5% extra credit will be given for especially thorough, well-documented, or insightful solutions.
- *** Where three '*'s appear in the homework, perform the required test(s) and turn in a printout of either:
  1. a table printed by your verilog testbench module listing all inputs and corresponding outputs,
  2. an Isim waveform plot which clearly shows (labeled and highlighted) corresponding inputs and outputs, or
  3. a section of testing code which clearly compares the designed circuit and a simple reference circuit, and two short cut & paste sections of text from your simulation (one for pass, and one for fail where you purposely make a slight change to your reference code to make it fail) that look something like this:

    Error: input=0101, out_module=11110000, out_ref=11110001

    In all three options, each test case must be marked whether the output is correct or not.

Keep "hardware" modules separate from testing code. Instantiate a copy of your processing module(s) in your testing module (the highest level module) and drive the inputs and check the outputs from there.
1. [75 pts] The purpose of this problem is to familiarize you with the synthesis and place and route process and to give you a rough feeling for the size of a few simple circuits. Turn in a detail report with the results for each part.

   o For all designs register the inputs and outputs, and add timing constraint for the clock (e.g. 550 MHz). Report slack (set-up and hold time if available). If it couldn’t make the timing constraint, slow down the clock until it meets the timing.
   o Synthesize and place and route the following blocks and report their total slice count, and other FPGA resource counts that are listed in Summary Report (this includes all logic utilization numbers under Device Utilization Summary). Include registers (flip-flops) and constrain the timing path as explained in the class. Assume words are all 2’s complement signed unless stated otherwise. No need to simulate, only turn in the source verilog, but your verilog must compile correctly (Use “Check Syntax” in Synthesis flow).
   o Report delay for each block using Place and Route Static Timing Analysis (under Place and Route). Slack must be a minimum positive number (zero or near zero e.g. 0.1 ns).
   o Using the XPower analyzer tool (under Place and Route), report the power dissipation of each block for your clock specification that the design can operate. Note that for power number, your design clk must be set to the number that you found in previous step, and place and route must be done again.
   o Report results for all circuits in a table. Report the numbers in a single table so it can be used as a note sheet in the future.
   o Include verilog files for each circuit in the report. Note there is no testbench needed for this problem and you don’t need to test the functionality of these blocks.

Blocks

   a) [10 pts] bitwise OR of two 10-bit numbers (10-bit output)
   b) [10 pts] 10-bit 3:2 adder (full adder) using verilog "&" ",|", ",^", ",~". Draw your circuit and the circuit output by Xilinx Synthesis tool (or by hand).
   c) [10 pts] 10-bit adder (11-bit output). Use "+" in verilog.
   d) [10 pts] an adder which adds 25 6-bit numbers using verilog "+" (i.e., something like, assign out = in0 + in1 + in2 + ...) and produces a 6-bit sum.
   e) [10 pts] 8-bit x 8-bit unsigned multiplier (16-bit output). Use "*" in verilog.
   f) [15 pts] 16-bit x 16-bit unsigned multiplier (32-bit output). Use "*" in verilog
      i. Redo the synthesis and Place and route steps by changing the Synthesize Options to use DSP-LUT block instead of Look-upDSP tables. Refer to the tutorial which is posted under the assignment.
Design a circuit which performs multiplication of two 2's complement unsigned integers sequentially. Construct the multiplier for two 246-bit operands containing just one adder which adds successive partial products over successive clock cycles. For more details refer to 2bit-mult-parallel-nopipeline.pdf which is posted as an additional file for the homework. This problem design consists of three major stages:

**Design**

i. [3 pt] Determine the width of the final product (i.e. how many bits the results must be).
ii. [7 pt] Draw a detailed data path block diagram
iii. [5 pt] Identify the control steps required for sequential multiplication using your datapath
iv. [20 pt] Develop a finite-state machine (FSM) for the control section. Assume that the X and Y operand values are valid on a cycle when a control signal, start, is 1. Generate a control signal, done that is 1 when the multiplication is complete. Report the control sequence and transition function based on the required steps in separate tables (as shown in the class), draw a state transition diagram for your FSM.

v. [20 pt] Develop the verilog model for both datapath and state machine (control section).

**Behavioral Simulations**

vi. [15 pt] Test the design and identify the extreme cases. Turn in *** Option 1.

**Implementation**

vii. [5 pt] Perform Synthesis and Place and Route and report total slice count, and other FPGA resource counts that are listed in Summary Report.

viii. [5pt] Report design speed using Place and Route Static Timing Analysis (under Place and Route). Slack must be a minimum positive number (near zero).

ix. [5 pt] Using the XPower analyzer tool (under Place and Route), report the power dissipation of the design for your clock specification that design can operate.