CMPE 415 - Homework #6

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. Design a circuit that compute the product of 128 element 16-bit vectors, a and b; that is a vector p such that \( p_i = a_i * b_i \). The elements of a and b are stored in separate SSRAMS and the result is to be written into a third SSRAM. Assume that computation is started by a control signal, “go” being one during a clock cycle and output control signal, done is to be set to one during the cycle when the computation is complete. Assume a and b are 16-bit values. You can assume data is already stored in ai and bi memories but for testing you need to write to the memories and test reading back for a few samples (e.g 10 samples).

**Design**

i. [10 pt] Draw the detailed block diagram with naming the signals

ii. [20 pt] Use a finite state machine for the control signals. 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.

iii. [20 pt] Write a verilog model for the SSRAM memories and write the verilog for the complete circuit.

iv. [15 pt] Design the circuit using Xilinx Core Generator to generate the memories as shown in the class. Instantiate the cores in your verilog design.

**Behavioral Simulations**

v. [20 pt] Test the design in part iii and iv and identify the extreme cases. Turn in *** Option 1

**Implementation**

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

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

viii. [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.
2. This problem was explained in the class: Design a FIFO to store up to 256 data items of 16-bits each, using 256x 16-bit dual-port SSRAM for the data storage. Assume the FIFO will not be read when it is empty, not to be written when it is full, and that the write and read ports share a common clock.

i. [20 pt] Develop a verilog model for the FIFO example.

ii. [20 pt] Test the design and identify the extreme cases. Turn in *** Option 1

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

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

v. [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.

vi. [30 pt] Develop the FIFO circuit using Xilinx Core Generator. Repeat steps ii to v.