ELEC 5250/6250/6256 Verilog Project #4

First draft – Due Wednesday, September 19 (for instructor feedback only – no grade)
Final version, including test bench and simulation – Due Monday, September 24

Design a hierarchical Verilog register-transfer-level (RTL) model of a circuit that performs the arithmetic division operation on two unsigned integer numbers. A is to be an 8-bit "divisor", and B is to be a 16-bit "dividend". The quotient, Q, and remainder, R, are each to be 8-bit unsigned integer values. The divide algorithm is to be an iterative "restoring division" or "non-restoring division" algorithm, using a sequence of 8 subtract/add and shift operations *(refer to various computer architecture text books for descriptions of restoring and non-restoring division algorithms.)*

1. Design four separate Verilog models: (1) register component (one common model to be instantiated for all registers), (2) the arithmetic unit, (3) the controller, and (4) a top-level model that instantiates these components. ***You may also need a multiplexer to select one of the register inputs. This may be designed as a separate component or implemented with a conditional signal assignment statement in the top-level design.***
2. Use "register-transfer-level" (RTL) design that primarily manipulates **vectors**, i.e. do not model individual gates and flip-flops in your components. A "register" should be a relatively simple design, with data inputs and outputs defined as vectors.
3. For the registers, design a **single** multifunction register (shift, load, etc.), and instantiate multiple copies of that as needed for each register in the divide circuit.
4. The arithmetic unit should be an RTL behavioral model, using arithmetic operators (add, subtract) between vectors; do not design a binary adder at the bit level.
5. The controller should be a behavioral model of a finite state machine.
6. The top-level should simply instantiate the registers, arithmetic unit, and controller. ***It may also include a multiplexer component or one conditional signal assignment statement to implement a multiplexer.***
7. **ELEC 5250** students: You may assume that only valid operands are to be divided (no overflow detection is needed.)
   ELEC 6250/6256 students: You are to detect "overflow" conditions – those that would generate invalid results (quotient greater than 8 bits, divide by 0, etc.) There should be an OVERFLOW output signal to indicate the occurrence of such a condition.
8. The functionality of each of the components should be simulated and tested individually, in addition to testing the top-level component. For the components, you may use either a "do file" or a test bench. For the top-level test, you are to use a "test bench" to exercise the circuit with at least 50 pairs of operands (which should span the range of possible numeric values), automatically check the result, and print a message if any errors are detected.

To be submitted:
1. The four Verilog models.
2. The top-level test bench.
3. An "edited" file of the top-level results from a **List Window**. Save the List Window as a plain text file, and then edit out all but a few lines at the start and conclusion of each of the 50 operations. **(Use minimal paper.)** *You do not need to submit simulation results for the three individual components.*