Department of Electrical and Computer Engineering
University of California, Davis

Lab 3: A Hierarchically-Built Sorter

I. Prelab

Complete the following and submit your work at the beginning of your lab session.

1. [10 pts] Complete and document a preliminary design of your circuit (a draft of your verilog code)

II. Project Overview

Custom digital hardware can have significant performance advantages over programmable processors because datapaths can be customized for specific calculations and also because of the ability to exploit a large amount of parallelism, workload permitting. This design is fully combinational meaning the entire process can be calculated in a single clock cycle enabling performance in a scaled-up design that is orders of magnitude faster than what a programmable processor can accomplish. Details not specified in this lab should be chosen by you and stated in your lab report.

Board-level input and output keys, switches and displays for the design are:

- KEY0 Selects whether HEX5–HEX2 show the sorter's outputs (KEY0 not pressed) or the sorter's inputs (KEY0 pressed)
- {SW1,SW0} Selects the input test number
- HEX0 7-segment display of input test number [3,2,1,0]
- HEX5–HEX2 Displays the four sorted outputs
- LEDR9 Lights when at least one input is 4'b1111
- LEDR8 Lights when at least one input is 4'b0000
- LEDR7 Lights when the four inputs are already in sorted order

All data words are 4-bits wide and unsigned (values range from 0 to +15). The four input data words are generated by the module testgen.v which is provided to you.

III. seg7.v

Design a module which takes in four input bits (in[3:0]) to be interpreted as a 4-bit unsigned number, and has a 7-bit output (out[6:0]) which drives a single 7-segment display. Inputs corresponding to values of ten to fifteen should appear as the letters A–F respectively.

Three of the outputs must be designed as `wire`s, and four as `reg`s. As much as possible, the method of your code should be different for each of the seven outputs (e.g., `if` versus `case`). Check the assertion level of the hex displays which are typically active low.

Hint: since an output port can be only a `wire` or `reg`, *out* must be a concatenation of the three `wire`s and four `reg`s.

Test your seg7.v design in simulation with a simple testbench.

Verify its operation on your FPGA board by connecting the inputs to SW3–SW0, perhaps with only a couple outputs implemented on the first test.

IV. sort2.v

Design a module which sorts two input values such that the lower value appears on output *out0* and the greater value appears on output *out1*.

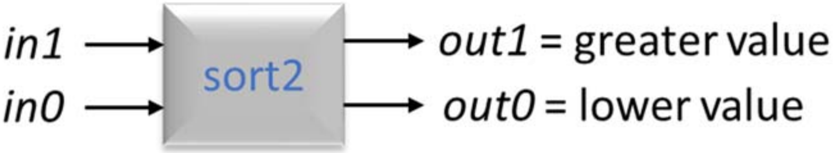

Figure 1. sort2.v module with 4-bit unsigned inputs and outputs

V. bubble4.v

Design a module which performs a single stage of a bubble sort as shown in Figure 2.

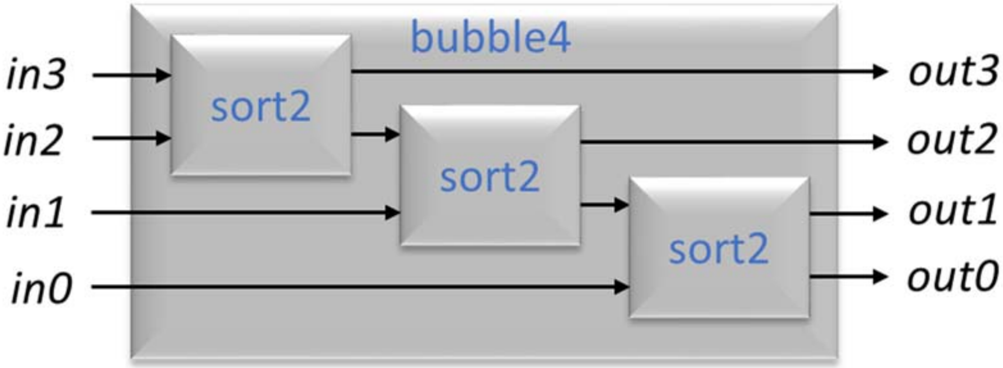

Figure 2. bubble4.v module block diagram

Test your bubble4.v design in simulation with a simple testbench.

VI. sort4.v

Design a module which performs a complete bubble sort of four inputs as shown in Figure 3.

Connect testgen.v to the inputs to provide test data.

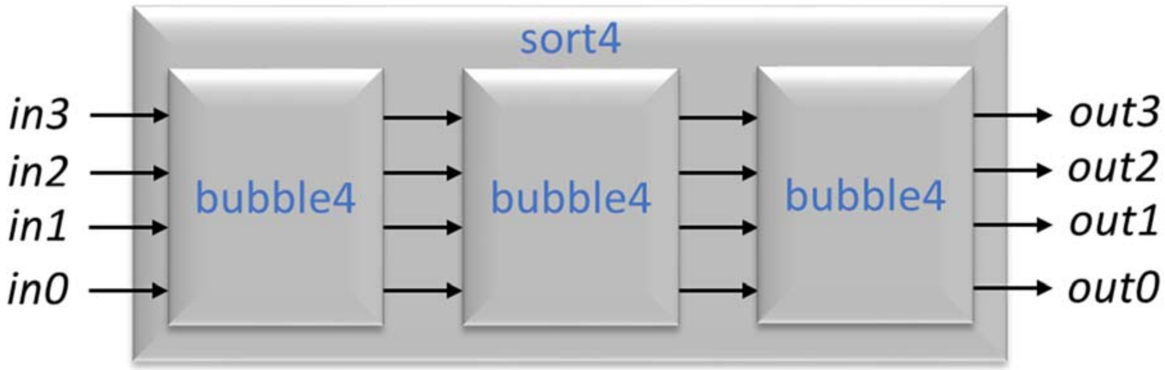

Figure 3. sort4.v module block diagram

VII. top.v

Instantiate copies of sort4.v, testgen.v, seg7.v, and other logic to implement the functions described in Section II.

VIII. Testing in Simulation

Design and write a testbench for your module that exercises all major functions.

Once your design is working correctly in simulation, demonstrate it exercising your module to your TA and have it checked off.

IX. Implementation and Verification on the DE10-Lite

Once your design is working correctly on the DE10-Lite FPGA board, demonstrate it exercising your module to your TA and have it checked off.

X. Timing Analysis

~~[15 pts]~~ Using the procedure outlined in "Tutorial: FPGA Timing" on the class website, determine and report the highest operating clock frequency for your design.

Submitted Work [100 pts total]

With the exception of instructor-provided code, all work must be yours alone.

[10 pts] **Prelab**

[75 pts] **Lab Checkoffs**

1) [25 pts] Demonstrate correct operation in simulation using your testbench

2) [50 pts] Demonstrate correct operation on the FPGA board

[15 pts] **Lab Report**

1. [5 pts] Report the maximum clock frequency your design could attain.

2. [5 pts] Calculate and report the number of input combinations required for an exhaustive test of:

   a) sort2.v

   b) bubble4.v

   c) sort4.v

3. Submit all Verilog hardware and testbench code that you wrote. Do not include any code that you did not write such as files generated by Quartus or IP components.

   A. [5 pts] Print and submit a paper copy during your lab session.

   B. Upload a copy to Canvas—this is essential to receive credit for the entire lab.

   Perform the following steps by the end of your lab session:

   1. Make a folder on your computer
   2. Copy all verilog files you wrote into the folder—only the ones you wrote
   3. “zip” the folder into a single .zip file
   4. Log onto Canvas, click Assignments, find the correct lab number
   5. Upload the .zip file

2018/04/24	Changed 7seg to seg7, added hint concerning seg7’s outputs
2018/04/23	Added note concerning draft code for prelab, added Section VII.
2018/04/20	Posted