EE 4702 **Homework 2 Due: 22 Feb 2001, 23:59:59 CST**

Solve this problem by modifying a copy of `http://www.ee.lsu.edu/v/2001/hw02.html` *(or .v) which can also be found in* `/home/classes/ee4702/files/v/hw02.v`. *See* `http://www.ee.lsu.edu/v/proc.html` *for instructions on running the simulator. Alternate instructions can be found in Lesson 7 of the ModelSim Tutorial, linked to the references web page,* `http://www.ee.lsu.edu/v/ref.html`. *The links are clickable when this assignment is viewed with Acrobat Reader. The ModelSim tutorial and other documentation can also be accessed from the Help menu on the ModelSim GUI (started by the command vsim -gui).*

In this assignment a priority encoder will be designed which has an n-bit input, called `request`, *and an n-bit output, called* `grant`. *Let bit positions be numbered from* $n-1$ *to* 0 *and let bit zero be the least significant and the rightmost bit when written. Output bit* i, $n-1 \geq i \geq 1$, *shall be* `1` *if input bit* i *is* `1` *and no lower-order bit, if any, is 1. Otherwise output bit* i *is zero. Therefore, at most one output bit is 1, corresponding to the first input bit that is* `1`. *Some examples:* $0011 \rightarrow 0001$, $0110 \rightarrow 0010$, $0111 \rightarrow 0001$, *and* $0000 \rightarrow 0000$, *where foo* $\rightarrow$ *bar indicates that output bar is expected for input foo.*

Problem 1: Complete the module `priority_encoder_8_b` so that it is a behavioral description of an 8-bit priority encoder as described above (and in Homework 1). Just include behavioral code, **do not** instantiate other modules.

Problem 2: Modify the `priority_encoder_1_es` modules from Homework 1 so that each gate has a delay of one cycle.

Problem 3: Complete the module `priority_encoder_8_es` so that it is an explicit structural description of a priority encoder constructed using `priority_encoder_1_es` modules from the previous problem. They may be instantiated within `priority_encoder_8_es` or you can provide an intermediate module, say `priority_encoder_4_es`, which instantiates `priority_encoder_1_es`.

Problem 4: Complete the module `test_pe_8` to that it tests `priority_encoder_8_b` and `priority_encoder_8_es`. Unlike the testbench in Homework 1, this testbench can be commanded to perform the test any number of times. Module `test_pe_8` has one input, `start`, and three outputs, `done`, `okay_b`, and `okay_es`. Initially, `done` should be 0. When `start` is 1 output `done` should be set to zero. At this point, no other outputs should change until `start` goes to zero. After start goes to zero the modules should be tested. When the tests are complete set `okay_b` and `okay_es` based on the outcome of the test. *After* setting `okay_b` and `okay_es` set `done` to 1. At this point, wait for `start` to go to 1 and repeat the process.

Use module `tests_pe_8` (two esses) to test the timing of your testbench. The testbench should be able to catch all errors, including undefined outputs.