

## ECE 368—Spring 2009, Instructor: Prof. Shantanu Dutt

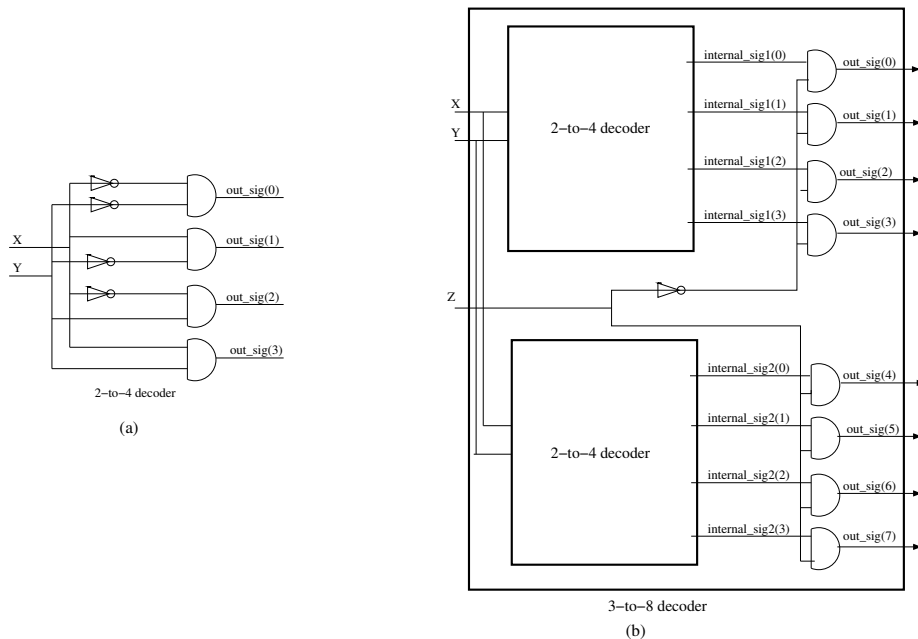
### Lab 1 : Due Thurs, Jan 29 (beginning of class)

## Goals

The goals of this lab are: (1) To give an example of the *divide-and-conquer (D&C) design method*—solving a design problem by breaking it into smaller problems and stitching up the solutions, the related *hierarchical implementation/coding method*—implementing a design using smaller components; (2) To make you realize the correct way to specify delays when assigning assigning different values to a signal under different Boolean conditions; (3) To understand what *glitches* are via the graphical simulator; (4) To determine delays via the graphical simulator; (5) To illustrate the use of a configuration description to ease the process of compiling and simulating different architectures of the same entity.

## Problem

The designs for a 2-to-4 decoder and a 3-to-8 decoder are given in the figures below. The 3-to-8 decoder is designed using a divide-and-conquer strategy that uses two 2-to-4 decoders and stitches them together using additional logic to yield a 3-to-8 decoder. Notice how clean the 3-to-8 decoder design is and imagine how much messier the design would be using a “flat” approach.



The VHDL codes for component AND and NOT gates (only *behavioral* architectures are described), the 2-to-4 decoder (three architectures, *behavioral*, *dataflow* and *structural* are described), the 3-to-8 decoder (two architectures, *structural* and *mixed* are described) and two

test benches one for the 2-to-4 decoder and one for the 3-to-8 decoder are given in the file <http://www.ece.uic.edu/~dutt/courses/ece368/labs/f05/lab1.tar>. Note that the coding for the 3-to-8 decoder uses a hierarchical approach in which the decoder is described in terms of smaller components like 2-to-4 decoders and logic gates whose descriptions are then coded. The two test benches have configuration descriptions that specify which architecture of their respective instantiated components is to be used.

Perform the following tasks.

1. Create a directory called "incorr\_delay\_decoders" under your src and sim directories. Store the vhdl codes under src/incorr\_delay\_decoders/ and compile them using vhdlan under sim/incorr\_delay\_decoders.
2. Run "scs configuration\_name" first for the test bench for the 2-to-4 decoder. Note that when configurations are described we need to invoke scs for that configuration. Test the behavioral, dataflow and structural descriptions of the 2-to-4 decoder by specifying the appropriate architecture in the test bench, recompiling it (vhdlan) and running scs on its configuration (i.e., "scs Config\_2to4"), and then running scirocco. Note that you need to run scirocco only the first time. For later invokings of vhdlan and scs, you can keep the various scirocco windows open and invoke ./scsim for running the new code.
3. For each run, display the inputs and outputs of the test bench as vectors IP and OP in Hex format (it is cleaner to see the display that way rather than see the binary values of each input and output bit).
4. For each run determine if you are seeing anything unusual and unexpected in the display and if so identify what it is in the printout of the display that you should put in your lab report.
5. Determine the delay specification problems, **if any**, in the VHDL codes of the appropriate files, correct them, put the corrected codes under directory src/correct\_decoders, compile them under directory sim/correct\_decoders, and rerun the codes (you need to rerun scirocco from the new directory sim/correct\_decoders) and check that that the unusual outputs other than *glitches* have been eliminated. Glitches at the output(s) of a circuit are temporarily incorrect output values produced for the current inputs before the correct values are established, and are not unusual.
6. Print the timing graphics and show where glitches are occurring and that the unusual outputs have disappeared. Determine the delays of the different architectures of the 2-to-4 decoder using the graphical interface and note them along with printouts of the timing graphics.
7. State in your lab report your conclusions on the correct way of specifying delays.
8. Repeat the above with the 3-to-8 decoder codes.
9. Submit your corrected codes along with your lab report.