University of California, Davis
Department of Electrical and Computer Engineering

EEC180B DIGITAL SYSTEMS II Spring 1999

Lab 4: Structural Modeling in VHDL

Objective: In this lab you will investigate the structural style of VHDL modeling. You will simulate and synthesize a hierarchical design and compare the synthesized design with a manual design.

Pre-lab: Use K-maps to derive the minimal algebraic expressions for Bout and D. This assignment must be turned in to your TA within the first half-hour of your first lab session.

Note: this lab and pre-lab are to be done individually. That is, you may not work with anyone else or use anyone else's work.

Problem Specification

In this exercise you will design a four-bit binary subtracter by using one-bit full subtracter cells. Each full subtracter cell performs the binary operation X – Y – Bin, where Bin is a borrow signal from the adjacent, less significant full subtracter cell. The outputs of the full subtracter are Bout, the borrow request signal for the adjacent, more significant cell, and D, the difference signal. The truth table for a binary full subtracter is given below:

X	Y	Bin	Bout	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

For reference material on binary subtracters, see section 20.3 in Fundamentals of Logic Design by Charles Roth, Jr.

Lab Requirements

1. Use K-maps to derive the minimal algebraic expressions for Bout and D. (Pre-lab)

2. Write the VHDL description of the full subtracter. Do not specify delays in your model since these cannot be synthesized.

3. Simulate your full subtracter using either a test bench or a simulation script. Print the waveforms from your functional simulation.

4. Synthesize the full subtracter for the class library. You will need to modify the class.scr synthesis script file to synthesize your design. Make sure you are using the .synopsys_dc.setup file for the class library. Print the gate-level schematic from the Synopsys Design Analyzer. Note: The Synopsys tools *are* case-sensitive so you must give the exact names of the source file, entity and architecture in the script file.

5. Write a structural model of a 4-bit subtracter using your one-bit full subtracter as a component.

6. Simulate your 4-bit subtracter using a test bench program or a simulation script. (Note: the test bench for a signed multiplier in Figure 4-14 is a useful example.) Make sure to test at least 10 different cases, including both positive and negative inputs and outputs. Verify that your subtracter is working properly. Print the waveforms from your simulation.

7. Synthesize your 4-bit subtracter for the class library and print the top-level schematic from within the Synopsys Design Analyzer.

   In order to synthesize a hierarchical design, each design module must be analyzed and elaborated in hierarchical order, beginning at the bottom of the hierarchy and proceeding to the top-level design module. An example synthesis script, sub4.scr, which synthesizes the 4-bit subtracter model into the class technology, is included at the end of this write-up. The script file is also available at /afs/ece/classes/eec180b/lab4/sub4.scr. Modify the design, entity, architecture, and output file names in the example script to synthesize your 4-bit subtracter.

Lab Report

Submit the following items in your lab report: (Due in your first lab period of the following week - i.e. one week after the start of the lab.)

- All VHDL source code.
- Test bench source code and/or simulations scripts.
- Gate-level schematic of your full subtracter printed from the Design Analyzer.
- Top-level schematic of your 4-bit subtracter printed from the Design Analyzer.
- Simulation waveforms from your functional simulations.
- Signed TA verification of your 4-bit subtracter simulation.
- Answers to the following questions:

    1. Using Boolean algebra, verify that the synthesized full subtracter circuit is equivalent to your input equations.

    2. Write a VHDL function to implement a 4-bit subtracter, following the example in Figure 2-22 of the text for the 4-bit adder. (Optional exercise – verify your function with a test bench program.)

```
/* ========================================================== */
/* CLASS library synthesis script file
/* ========================================================== */

/* To execute this DC Shell script, from command line, type:        */
/*        dc_shell -f sub4.scr > sub4.log &                         */

/* Source file, entity and architecture of the lower-level (1-bit) module.     */

comp_name = fullsubtracter.vhd
comp_ent = fullsubtracter
comp_arch = equations

/* Source file, entity and architecture for the top (4-bit) module.     */

design_name = sub4.vhd
ent_name=sub4
arch_name=structure

/* Note that output file names must be all lower-case for           */
/* compatibility with the Xilinx tools.                             */

output_name=sub4_class

/* Analyze and elaborate design files in hierarchical order.        */
/* I.e. Modules which are referenced in other files first.          */
/* Packages, low-level modules, finally the top-level design.       */
/* Analyze the source file.                                         */
/* Elaborate the entity, specifying the architecture.               */

remove_design -all

analyze -format vhdl -lib DEFAULT {comp_name}
elaborate comp_ent -arch comp_arch -lib DEFAULT -update

analyze -format vhdl -lib DEFAULT {design_name}
elaborate ent_name -arch arch_name -lib DEFAULT -update
current_design ent_name
link
set_operating_conditions -library "class" "WCCOM"
set_wire_load  "10x10" -library "class"
uniquify
compile  -map_effort medium
write -format db -hierarchy -output output_name + ".db"
report_area > output_name + ".area"
report_timing > output_name + ".timing"
quit
```