EECS 303: Advanced Digital Logic Design
Lab Four - Sequential FSM design in VHDL

Robert Dick

Assigned: 20 November
Due: 2 December

1 Introduction

In this laboratory assignment, you will specify a finite state machine at a fairly high level in VHDL. You will then simulate the machine using ModelSim from Mentor Graphics. After simulation, you will use Synopsys Design Compiler to synthesize it, automatically converting it into an implementation composed of basic memory elements and gates.

2 Tutorial

In this section, I'll walk you through the specification, simulation, and synthesis of a sequential machine that that accepts all sequences ending in 0, 0, 1.

You don't need to hand in anything for this portion of the laboratory assignment. However, you will need to use what you learn by following the tutorial to complete the assignment in the next section.

1. Take a look at the "`/vol/ece303/recog.vhdl`" file. This file contains a VHDL specification for the state machine accepting (0 + 1)001. It defines a pattern recognizer with three input ports: a clock signal (clk), an input (a), and an asynchronous reset (reset).

   The STATE process describes when the state variables change. Whenever reset goes high, the machine is forced into state *s0*. When a rising clock edge is detected, the machine's state variables are changed to the next-state values. The NEW_STATE process describes the next state as a function of the current state and input. The OUTPUT process describes the system output as a function of the current state (Moore machine).

2. Take a look at the "`/vol/ece303/test-recog.vhdl`" file. This file contains a test bench for the RECOG entity. The CLK_CHANGE process generates a clock signal with a period of 10 ns. The INPUT_CHANGE process applies an input sequence of 010010110001.

3. Make a working directory, move into it, and copy the example vhdl files to the directory

```
mkdir ece303-lab4
cd ece303-lab4
cp /vol/ece303/recog.vhdl .
cp /vol/ece303/test-recog.vhdl .
```

4. Set up your environment by executing the following commands

```
source /vol/ece303/mgc.env
source /vol/ece303/synopsys.env
```

5. Link the Synopsys Design Compiler configuration file to your own directory.

```
ln -s /vol/ece303/synopsys_dc.setup ~/.synopsys_dc.setup
```

6. From within your work directory (ece303-lab4), type

```
qhlib work
```

   to create a work library in which parts of your compiled design will be placed.

7. Compile the VHDL files for the recognizer and test bench

```
qvhcom recog.vhdl
qvhcom test-recog.vhdl
```

   You shouldn't get any errors.

8. Simulate the design.

   (a) Start Mentor Graphics ModelSim

   ```
   qhsim
   ```

   (b) Close the annoying "New ModelSim Features!" window.

   (c) Click on the "`+`" to the left of "`work`" in the window to the left.

   (d) Double-click on "`test_bench`".

   (e) Select "`View`→`Signals`" from the menu. A new window will appear containing the signals available in the test bench: clk, input, reset, and output.

   (f) Select "`Add`→`Wave`→`Signals in Region`" from the signals window menu. A new window will appear showing the signals.

   (g) In the wave window, use the LMB to drag the line to the right of the signal names to the right so the full signal names are visible.

   (h) In the ModelSim window, select "`Simulate`→`Simulation Options`" from the menu. In that window, set the default run time to 150 ns. Then click "`OK`" in the Simulation Options window.

   (i) Select "`Simulate`→`Run`→`Run 150 ns`" from the ModelSim window. Now you have some waveforms in the wave window.

   (j) Select "`View`→`Zoom`→`Zoom Full`" from the wave window menu to get a better view of the waveforms.

   (k) To print the file, you can go to the "`File`→`Print PostScript`" in the wave window menu. Change the print command from "`lpr -d lp1`" to "`lpr -P laser1`". If you want, you can print to a file, instead.

   (l) You can shut down ModelSim now. However, if you found problems in your own design and want to get more detailed information, you can go back to the signals window and add all the signals in the design to the wave window. The "`Simulate`→`Run`→`Restart`" option in the ModelSim window will be useful to reset time.

9. Synthesize the design.

   (a) Start up the GUI for Synopsys Design Compiler

   ```
   design_analyzer
   ```

   (b) Select the LSI target library by selecting "`Setup`→`Default`".

```
Link lib:  * lsi_10k.db
  Target:  lsi_10k.db
  Symbol:  lsi_10k_sdb
```

(c) Compile the design by selecting “`File`→`Analyze`” from the menu. If necessary, select the “`recog.vhdl`” file from the list. Do not attempt to compile the test-recog.vhdl design. You only want to build hardware for the machine, not its tests.

(d) Synthesize the design by selecting “`File`→`Elaborate`” from the menu. Select the “`DEFAULT`” library and the “`RECOG(STATE_MACHINE)`” design from the lists. You will now have an image of the design in the Design Analyzer window.

(e) In the Design Analyzer window, double-click on the RECOG block.

(f) Double-click on the block again to get an un-optimized view of the design.

(g) Print the design by selecting “`File`→`Plot`” from the menu. Change the “`lpr -Plw`” command to “`lpr -Plaser1`”.

(h) Optimize the design by selecting “`Tools`→`Design Optimization`” from the menu. Tell it to use high effort when mapping.

(i) Print out the optimized design.

3 Your assignment

Based on your experience with the tutorial, design and implement a sequential system to solve the following problem.

1. Design a single input sequential synchronous machine with two outputs, L and M. L should be 1 if and only if input sequence during the last two cycles was 01. M should be 1 if and only if the input sequence during the last three cycles was 101. The machine should have an asynchronous reset input.

   You may produce a modified (multiple types of accept states) NFA to help you, or go straight to a FSM. I recommend using an NFA. My multi-output NFA has six states. My multi-output DFA also has six states. Hand in a state diagram for your FSM.

2. Describe the FSM in VHDL. Hand in your VHDL code.

3. Design a test bench subjecting it to the following input sequence: 100101101001101. Use a clock with a period of 10 ns. Simulate the design using ModelSim. Hand in your testbench VHDL code and a printout of the simulation waveforms.

4. Synthesize the FSM using Design Compiler. Hand in a printout of the unoptimized and optimized circuit diagrams.

5. Write six or fewer sentences discussing how long it would have taken you to do this design by hand.