CSE 260 – Digital Computers: Organization and Logical Design

Lab 4

Doug Shook *Due 11/17/2015*

Introduction

In this lab you will implement a stack using VHDL. You will then implement your stack on an FPGA.

1. Design

We will be implementing a stack. A stack works as follows:

Our stack can hold up to 16 bits. When the stack is first created (or reset) it should be empty. A stack has four inputs: push, pop, dataIn, and reset. If the push signal goes high, the value of dataIn is read and stored on the top of the stack. If the stack is full, then this value is ignored. When the pop signal goes high, the value on top of the stack is removed from the stack, and the value that was second from the top becomes the new value on top of the stack. If both push and pop are high, you should treat it as a push (in other words, ignore pop if push is high). This operation is called Last in First out (LIFO).

The below grid represents a new, empty stack:

Here is what the stack will look like if “push” goes high when “dataIn” is also high:

															1

If we continue to push a “0”, then another “0”, then a “1”, then finally a “0” the stack will look like this:

											1	0	0	1	0

If we set “pop” to high, then our stack will look like this:

												1	0	0	1

Our stack will have three outputs: empty, which is high if the stack has nothing in it, full, which is high if the stack is full (16 bits), and dataOut, which is the value that is currently on the top of the stack.

Your first task is to construct a high level state machine that implements a stack.

You should next sketch the circuit for your datapath (see section 5.6 of the ZyBook). You do not need to sketch a circuit for the HLSM, though you may if you wish.

2. Implementation

Create a new project and module and implement your stack using VHDL. Some things to note:

* To represent an empty position in your stack, you should use “X”, which stands for a “don't care” bit. You can assign “X” to std_logic or std_logic_vector just like any other assignment. This will ensure that the stack will display properly on the FPGA. If this is giving you trouble during testing/demoing, then it is acceptable to use “0” to represent empty positions as long as everything else works properly.

* You are not required to explicitly implement states in VHDL, but you may if you wish.

* Your VHDL needs to match the datapath exactly. We will be checking for this when we grade your lab.

* If the stack is empty, the value of dataOut does not matter and will be ignored.

3. Testing

Once you have finished writing code for your FSM, you need to create a testbench for it. Your testbench needs to test every possible combination of outputs. There are 3 single bit outputs, and 5 possible output combinations. Once you are satisfied with your testbench and simulation results, you should print out a copy of your simulation. You should then annotate your simulation printout with each transition that takes place.

4. Hardware Implementation

After successfully simulating and testing your circuit, you should implement it on an FPGA. This means you mus complete a top.vhd module that connects your module to the lcdDisplay, buttons, knob, and LEDs on the FPGA. Download these files and add them to your project:

http://cse.wustl.edu/~dshook/cse260/labs/lab4.zip

The only file you should modify is top.vhd. Comments have been included in this file that tell you exactly where you need to add or change code. You need to do the following:

* Instantiate your stack component.

* Create create a signal for your “dataIn” signal and connect it to inputs so that it can be controlled by the knob.

* Connect push, pop, and reset to buttons on the FPGA.

* Create signals for your outputs (empty, full, dataOut) and connect them to the *outBits* signal.

Once you have finished modifying top.vhd, you should deploy your project on the FPGA just as you have done on previous labs. You should see the input on the top row of the lcdDisplay, controlled by the knob as in the past. The contents of your stack should be on the bottom row of the lcdDisplay. You should be able to control the stack with the knob and see the results appear on the lcdDisplay as well as the LEDs.

What to Turn In

You must turn in a hard copy of the following:

* Your high level state machine.

* Your datapath circuit.

* An annotated printout of your simulation, as described above.

* A completed demo form (see below).

You must turn in a digital copy of the following (on Blackboard):

* Your VHDL code for the finite state machine.

* Your VHDL code for the testbench.

* Your completed top.vhd module.

Demo

You must demo your lab on an FPGA before you turn it in. Print out a copy of the demo form from the course website and hand it to the TA. They will walk you through the demo, fill out the form, and then give it back to you. Staple the completed form to the front of your report when you hand it in.