CS 6501 FPGA Familiarization Assignment

The goal of this assignment is to give you an overview of the steps involved in programming an FPGA. As a simple, illustrative case, you will program the FPGA to detect a specific 4-bit sequence (1011, left to right) from streaming data. The streaming data will be simulated by manual pushing of buttons on the FPGA board. The sequence consists of 4 bits of information. The LEDs on the board will represent the level of matching (1 LED means one bit matches, 2 LED's means 2 bits match …etc). For those familiar with the game Mastermind, you can imagine how to extend this project to fully implement that game, and if you are interested, you are encouraged to try! (But we'd rather have a correct implementation of the required task than an incomplete implementation of something more sophisticated.)

Example:

Input	1	0	0	1	1	0	1	1	0
LEDs	1	2	0	1	1	2	3	4	2

a) RTL Design

First, design a State Diagram for the Moore Machine. Then, write a Moore state machine to detect the sequence. You can use VHDL if you are first to RTL design or Verilog if you are familiar with. (Look at the "1.3 Concepts of VHDL" from the tutorial link and look at the examples link to learn about the syntax of VHDL)

b) Simulation Test

Simulate the previous example with your code using Modelsim (simulation tutorial)

c) RTL and Technology schematic

In the processes window (on the left), you can select Synthesize -> View RTL Schematic or View Technology Schematic. You can see how the tool creates the logic circuit of your code. Compare the number of blocks and logic gates in the "RTL Schematic" view to the "Technology Schematic" once which is a translation of the RTL schematic into the FPGA available components. Additionally, submit one of the K-Map of one of those LUT (double click on the LUT in the "Technology Schematic view" to see its properties)

d) FPGA place and route view

In the processes window (on the left), you can select Implement Design -> Place & Route -> View/Edit routed Design (FPGA Editor). This will show you how your code is placed and routed into the FPGA. Looking at that screen, submit the count of slices used in your design (each group of blocks is 1 slice)

e) FPGA Test

Once you are sure of the functionality of your code generate the "bit" file and install it to the FPGA board in Lab 320. The board is set up with the right version of Xilinx in the lab. You can check the power point presentation for the steps to download the program to the FPGA.

Use the I/O link to specify the buttons and the LEDs on the board which represent the input and output (choose two buttons to represent 0 and 1).

Submit:

1. Your state machine diagram
2. Commented vhdl/verilog code and the .ucf file which contains I/O assignments
3. Screenshot of simulation of the project showing the mentioned example (use Modelsim or any other simulator)
4. Comparison between the number of blocks in the RTL schematic and the Technology Schematic
5. Submit a picture the Karnaugh Map of one of the "Look Up Tables" by checking the Technology Schematic view.
6. Number of Slices used in your project from the Place & Route view (each spatially grouped components is considered 1 slice)
7. Picture of the Board with all 4 LEDs lit (if possible)
8. Small write up describing your approach and any interesting points came up while doing the homework.

Hints:

Hint 1: Push buttons and LED are active low

Keep in mind, the buttons & and the LED's are ACTIVE LOW. Which means the button will be sending '1' when not pressed and when pressed it will send a '0'. Same thing with LED, if an LED is assigned to 0, then it will light, if it is assigned to '1' it will turn off.

Hint 2: Debouncing push buttons

To de-bounce the buttons, use the following code as the clock (remember this is not an automated clock signal, it is a manual input signal which I called clk).

```
clk <=      '0' when  clk_low = '0' and clk_hi = '1' else
            '1' when  clk_hi  = '0' and clk_low = '1';
```

This code will simulate a clean signal "clk" using two bouncy buttons "clk_hi" & "clk_low". Each button will represent a rising edge and a falling edge. Use that signal to indicate new bit.

Hint 3: Version compatibility

You can write your VHDL code on any Xilinx version, but to generate the .bit file you need 10.1 .Other versions doesn't support the available FPGA. The computer in the 320 lab has the right version of xilinx and modelsim already installed.

Warning:

The computer in the lab DOES NOT HAVE internet. Keep that in mind when trying to transfer files from and to it

References and tutorials

- VHDL language tutorial: http://www.vhdl-online.de/tutorial/
- VHDL Examples: http://www.vhdl.org/vhdlsynth/vhdlexamples/
- Simulation tutorial: http://www.cis.upenn.edu/~milom/cse372-Spring06/simulation/
- Modelsim power point: [See attachment to assignment page in Collab]
- I/O: http://www.stanford.edu/class/ee108b/labs/ug069.pdf (page 35)
- Bit generation power point: [See attachment to assignment page in Collab]