CprE 583 (Fall 2009)
MP0: Testing out tools (Echo serial port)
(Due Friday: 9/11, Midnight)

I. Download and file locations

1. Download MP0.tar.gz from http://class.ece.iastate.edu/cpre583 into you ISU home directory (also refered to as your ISU U: drive)

2. Uncompress the assignment
    a) gunzip MP0.tar.gz
    b) tar -xvf MP0.tar

3. ISE project is located here: MP0/MP0_ise.ise
    a) Set up environment
        i. Make sure you are using a bash shell. "echo $SHELL" to check. If you are not then type "bash" to get into a bash shell
        ii. Note: you can set your default shell to bash at https://weblogin.iastate.edu/cgi-bin/index.cgi (Manage User -> View/Edit your Linux login shell -> select /bin/bash
        iii. source /usr/local/bin/Xilinx_src
    b) To open project: ise MP0_ise.ise &

4. You should be able to edit your VHDL, and build you bitfile from ISE

5. Viewing the Hardware (Distance Students), on campus students can used the two machines in the back of Coover 2041:
    a) http://xilinxcam1.ece.iastate.edu: Is used to see the LEDs
    b) http://xilinxcam2.ece.iastate.edu: Is used to see the Monitor (Not needed this lab)
    c) username: xilinxuser
    d) password: drjones
    e) Distance student hardware access
        i. Use NX to login to xilinx.ece.iastate.edu
        ii. Note using NX is highly recommended. ssh -X will be WAY too slow
    f) NX download locations
        i. For Windows: http://www.nomachine.com/download-client-windows.php
        ii. For Linux: http://www.nomachine.com/download-client-linux.php
        iii. For MAC OS: http://www.nomachine.com/download-client-macosx.php
        iv. For Solaris: http://www.nomachine.com/download-client-solaris.php

6. Sharing the Hardware
    a) Everyone should do the following before and after using the hardware on xilinx.ece.iastate.edu (Note: Local students should be primarily using the on campus resources)
        i. Check if anyone else is using the ML507.
        ii. If no one responses to your request, then announce you will be using it. Maybe even give a time frame for when you will be done.
        iii. Send an message when you are finished

    b) Two commands that we will use for this are "mesg" and "wall".

mesg:

mesg: allows others to write to your terminal

mesg without any options tells you if others can write to you (y/n)

mesg y: Allows others to write to you.

mesg n: Disables others from writing to you.

Make sure you type "mesg y" before you start using the HW, so you can tell when others request it. Also type "mesg" to verify that it does return "y"

wall:

wall allows you to broadcast a message to everyone logged into the same computer (e.g. xilinx).

Example:

wall Is anyone using the HW?

wall OK, I have not heard from anyone so I am going to use it from 7 – 7:15 pm, I will let you know if I finish earlier.

wall I am finished with the HW.

If I have time to think of, and more importantly implement a better way to share the hardware, then I'll let the class know. And if anyone has any simple ideas for sharing the HW please let me know.

    c) Please try to be aware of how loaded a given machine. You can use "who", "ps – au" and "top" to get a feel for how loaded a machine is. Also over the next week we should be adding 2-3 more machines for remotely accessing the Xilinx tools. I also plan to add one more ML507 for the distance students to primarily access

II. Getting started

1. Before you make any changes, make sure the base design works.
    a. Build the bit file within ISE
        i. In the “Sources for” window make sure “Implementation” is selected
        ii. In the “Sources for” window highlight MP0_top
        iii. In the “Processes for MP0_top” window
            1. Right Click: Generate Programming files
            2. Select “rerun all”
    b. Load bitfile to the FPGA
        i. Type “impact &” at the command line prompt
        ii. Create new project
        iii. Configure Device using boundary scan, Auto connect
        iv. Bypass until you get to the “fx70” this is you device
            1. Select MP0_top.bit
            2. Open
            3. OK
        v. Right click on the “fx70”
        vi. Program, select Pulse PROG, OK
        vii. Bit file should load. You should see the DONE LED turn off, then turn back on GREEN indicating the FPGA was successfully programmed (Note the DONE LED is two away from the blinking RED LED).
    c. Start minicom in a new terminal window
        i. minicom
        ii. Make sure it is set to 9600 8N1
            1. Ctrl-A Z
            2. Configure Minicom: O
            3. Serial port setup
            4. Bps: E
            5. 9600: E
            6. Enter, Enter
            7. Exit
    d. When you type you should see characters appear in the minicom window. If you do not then you should contact me.

III. Modify the VHDL to convert a-z to (A-Z)

You will modify MP0.vhd to convert the letters a-z to A-Z when you type on the keyboard. In other words when you type an "a" you will see "A" appear in minicom. This should not take more than about 5 lines of VHDL. Hint: goto http://www.asciitable.com/ an look at the Hex (Hx) encoding of the characters a-z and A-Z.

1. Within ISE open MP0.vhd
    a. In the "Sources for" window right click on MP0_top and select "open"

2. Look through MP0.vhd and I recommend at the same time looking through one or both of the following tutorials (or find one via Google that you like), and try to get a feel for what MP0.vhd is doing.
    a. http://www.seas.upenn.edu/~ese201/vhdl/vhdl_primer.html
    b. http://www.vhdl-online.de/tutorial/

3. Simulate the design. In general you should ALWAYS simulate your VHDL to verify that its logic is correct. In this case you should find that MP0.vhd is echoing data back to the driver.
    a. In the "Sources for" box select "Behavioral Simulation"
    b. In the "Sources for" box highlight MP0_tb (this is the testbench
    c. In the "Process for: MP0_tb" box expand "ModelSim Simulator"
        i. Right click and select "Rerun all"
    d. ModelSim will start up an begin to simulate your design
        i. In the lower left hand corner you can see how long it has simulated for. I have set the default to simulate for 10 ms. (Should take 1 – 3 minutes to complete simulation.
        ii. Once the simulation is finish click in the wave window
        iii. Click Zoom full (this will show the results from t=0 to t=10ms)
    e. Now use your VHDL to figure out which signals to take a closer look at in order to verify if your design (MP0.vhd) is working as expected
    f. Also look at the testbench driver .vhd file (MP0_top_driver) to see if you can use the simulation to help you understand what the VHDL design is doing.
    g. Refer to lecture 3 for a reminder of how to navigate within the simulation and ISE

4. Between the lines "Your code below here!!!!" and "Your code above!!!" place your VHDL code to convert a-z to A-Z. (Don't forget to save your changes)

5. Once your VHDL design appears to be correct in simulation, try to build a bitfile to down load to the HW (follow the directions from II. Getting Started).

IV. Explore the design in more detail.

Get practice using ISE and Modelsim. Explore more details of the design. The core of this MP is built on top of a UART design available at: http://www.mitzanu.ro/vhdl/index.html and is meant to be a very clean and simple implementation of a UART core.

1. Add signals for some lower level components of MP0.vhd
    a) In ModeSim in the Workspace window expand the hierarchy of MP0.vhd (uart_hw), and start adding signals to look at.
    b) To add signals to the waveform
        i. Highlight a component in the Workspace window
        ii. Select the signals that pop up in the Objects window (these signals will match what is in the VHDL code)
        iii. Drag and drop these signals into the wave window
        iv. Format the signals appropriately, and add dividers to label groups of signals. Note adding dividers is VERY important for organizing groups of signals. Right click in the wave window for formatting options, and adding divider options.
    c) Save your new wave file. If you do not save you will need to Re-add and reformat your signals if you restart the simulation. It is a good idea to get into the habit of saving you wave file/format often. You will find it is VERY frustrating to do LOTs of formatting, just to have to redo it because you forgot to save.
        i. In ModelSim click in the wave window
        ii. File->save
        iii. Browse
        iv. Select "MP0_wave.do"
        v. Save (if asked if you want to overwrite say "yes")
2. Running simulation for more time.
    a) In the Transcript window (in ModelSim)
    b) run <time> <unit>, For example: run 1 ms

V. What to turn in.

1. Email me your modified MP0.vhd (Both on campus and Distance students)

2. A copy of your wavefile format and dataset showing that your modified MP0.vhd works in simulation (see FAQ for how to save your waveform dataset, see IV.1.c for how to save your waveform format).
    a. MP0_wave.do
    b. MP0_wave.wlf
    c. Specify a time stamp and group of signals I should view.

3. On campus students: Demo your working bitfile to me in Cover 2041.

VI. Updates and FAQs

1. How to save a simulation/waveform dataset
    a. Dataset
        i. File->Datasets->(highlight vsim.wlf)->Save As->name_file.wlf-> Done
    b. Waveform format
        i. File -> Save Format -> name_file.do

2. How to load a data set and waveform format (Note load dataset first)
    a. Dataset
        i. File Datasets -> Open -> Browse -> (select wanted .wlf file)
    b. Waveform format
        i. File -> Load -> (select wanted .do file)

3. How to unlock the cable

There are two types of locked cable issues:

1. The locked cable will not let you download a bitfile

Two solutions (if neither works reboot the machine, for distance students send an email asking for the machine to be rebooted)

Solution 1:

```
impact -batch
setMode -bscan
cleancablelock
quit
```

Solution 2:

```
xmd
xclean_cablelock
```