EEL4713 Assignment #4
Spring 2012
Assigned: 2/9/12
Checkpoint Demo: 2/23/12 in Lab
Demo: 3/15/12 in Lab
Due: 3/17/12 @ 11:55pm Via Sakai

Section 1: Setup
There is no setup in this assignment.

Section 2: Textbook Questions
There are no textbook questions for this assignment.

Section 3: Laboratory
In this laboratory you will be designing a single cycle implementation of a MIPS processor. Your processor will need to be able to successfully execute 29 instructions – all the instructions listed under Core Instruction Set on the green reference sheet from your book, except for ll (load linked) and sc (store conditional). Please read sections 4.3 and 4.4 of your textbook for insight on how this can be done.

Begin your design by implementing one of the 29 instructions in a fashion similar to figure 4.17 in your textbook. Then, modify your previous hardware to implement the next instruction, adding any additional control signals needed for the instruction to function correctly. Repeat until all 29 instructions are fully implemented. Your report should include a separate section for each instruction with the following:

- A discussion of the modifications to the previous instructions hardware needed to implement the current instruction (including a screenshot or hand drawing of the added hardware)
- The control signals you had to add and all of the values the control signals must have for the instruction to operate correctly
- An annotated simulation output that proves that the instruction works correctly
- Timing information that describes how adding this instruction affects the previous clock rate.

If an instruction can be implemented with the previous instructions hardware simply say so but still discuss the values the control signals must have for the instruction to operate correctly and provide an annotated simulation output that proves that the instruction works correctly. This is the most important part of your report so please take it seriously. Without this part of your report I have no way of knowing if you implemented each instruction correctly; more explicitly, without this part I have no way of giving you partial credit if your entire processor does not function perfectly.

Checkpoint: Implement at least one R-Type and at least one I-Type instruction and demonstrate their functionality no later than the checkpoint demo date on 2/17/11.

Once you are sure that all 29 of your instructions work correctly, implement the control for your processor that takes the instruction opcode as its input and outputs all of the control signals that you created. Begin your design by modifying the table you created in assignment 3 so that it includes the instruction opcode and the control signals that are true during the instruction. Once you have modified the table it should be easy to implement a device that outputs the correct control signals based on the opcodes listed in your table. Combine your datapath, memory, and control into a single component called "mips_core". The system should have only three input signals, the system clock "clk", the memory clock "mclk" that operates at three times the rate of the system clock, and an asynchronous reset "rst". The reason for using two clocks is the limitation of Cyclone family of FPGA's, which only allows for a synchronous implementation of the RAM. Compile, simulate and test your design with the test program that will be posted on the website (lab4_test.mif).

For the DEMO, a demo program will be posted on the website the day of the DEMO (lab4_demo.mif). Include a section in your report where you decode both the test and demo programs into MIPS assembly code along side the functional simulation output (in table form) of your processor. You may consider writing a program or script in your language of choice to make disassembly easier. Make sure you annotate your output to show that your processor executed the program correctly.

Provide a detailed diagram of all interconnects in your system (similar to the figure 4.17).

Timing analysis of your processor should be done with the CycloneII EP2C8T144C8 FPGA (which is the FPGA in you EEL4712 board if you want to test your design on actual hardware) as your target device. Part of your DEMO grade will be based on the maximum clock rate listed in the timing analyzer report. The points will be assigned on a linear scale for frequencies ranging between 25MHz and 45MHz. Frequencies above 45MHz will receive extra credit. Anything below 25MHz will receive a zero. In your report make sure to discuss what design choices led to your clock rate and what can be done to improve the speed of your processor.

Below is a list of things to keep in mind while designing your processor:

- Your instruction memory should be implemented with an "altsyncram" component, a 32-bit output bus, an 8-bit address bus, a 256 word capacity, and mapped to the memory block beginning at address 0x00400000.
- Your data memory should be implemented with an "altsyncram" component, a 32-bit input/output bus, an 8-bit address bus, a 256 word capacity, and mapped to the memory block beginning at address 0x10000000. Your data memory should also have ram enable, write enable, and byte enable signals.
- Your processor should utilize a Little-Endian memory scheme.
- Register zero should always have the value zero so you need to alter your register file from assignment 3 so that register zero cannot be modified.
- Because the instruction memory is mapped to the 256 word memory block beginning at address 0x00400000, if you use your reg32 component from assignment 2 for your program counter you will need to modify it so that the value of the program counter is 0x00400000 after a reset.
- The shift instructions shift the contents of Rt not Rs as the green reference sheet suggests.
- The load instructions are I type instructions and therefore do not have an opcode of zero as the green reference sheet suggests; what the sheet lists as the function code is actually the opcode.

This list will be updated as questions arise.

Note: in addition to submitting your reports via Sakai, you must demonstrate your components functionality in Lab on the DEMO dates listed on the first page of this Assignment. You also need to submit your design files electronically by Zipping them before uploading to Sakai. Failure to submit your design files will cause you to receive a zero on the design sections tab of the Assignment. Please make sure that you only submit your VHDL code, BDF files, mif files, and/or symbol files and don't send any other files generated by Quartus.