EEL4713 Assignment #2
Spring 2012
Assigned: 1/19/12
Due: 1/28/12 @ 11:55pm Via Sakai

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

Section 2: Textbook questions
Questions from chapter 2: 2.11.4-6, 2.14.1-3, 2.21.1-2

Section 3: Laboratory
In Assignment 4 you will be asked to design a basic single cycle microprocessor that supports the MIPS core instruction set (29 instructions) listed on the front of the green reference sheet that came with your book. In this and the next assignment you will be asked to design several components that you will need to accomplish this task.

To measure the propagation delays, use the Stratix II EP2S130 FPGA device

There will be situations in the design process where you will need to select between two 1-bit signals. In order to solve this problem you will design a 1-bit multiplexer (name your VHDL entity "mux1") that will select between two 1-bit input signals, "in0" and "in1", depending on the value of a third 1-bit select input signal. This third signal, "Sel", should cause an output signal, "O", to take the value of "in0" when Sel=0 or the value of "in1" when Sel=1. Compile, simulate and test your design; turn in the VHDL entity and architecture, and a screen capture of a simulation trace showing the multiplexer's output for all possible input combinations.

Question 3.1: What is the worst-case propagation time from input to output of your 1-bit multiplexer?

If you look at the ***BASIC INSTRUCTION FORMATS*** section of your green reference sheet you will see that the values rs, rt and rd, each corresponding to 1 of 32 distinct registers, are represented by 5-bit numbers. During the instruction decode process it is sometimes required to select between these 5-bit values. For this reason you will alter your 1-bit multiplexer design to create a new component "mux5" that will select between two 5-bit input signals, "in0" and "in1", depending on the value of a third 1-bit select input signal. This third signal, "Sel", should cause an output signal, "O", to take the value of "in0" when Sel=0 or the value of "in1" when Sel=1. Compile, simulate and test your design; turn in the VHDL entity and architecture, and a screen capture of a simulation trace showing the multiplexer's output for two distinct input words and both sel=0, sel=1.

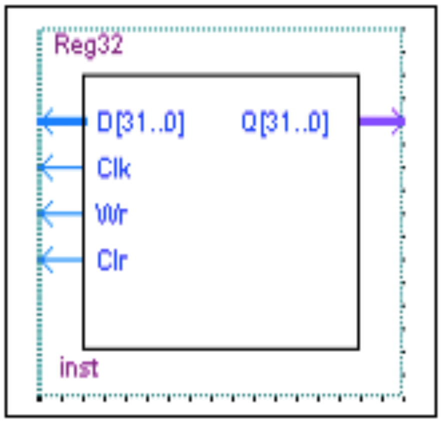
Question 3.2: Why are the values rs, rt, and rd represented with 5-bit numbers?
Question 3.3: What is the worst-case propagation time from input to output of your 5-bit multiplexer?

The MIPS processor you are designing is a 32-bit processor. Because of this there will be many situations where you will need to select between two 32-bit numbers. Alter your 1-bit multiplexer design to create a new component "mux32" that will select between two 32-bit input signals, "in0" and "in1", depending on the value of a third 1-bit select input signal. This third signal, "Sel", should cause an output signal, "O", to take the value of "in0" when Sel=0 or the value of "in1" when Sel=1. Compile, simulate and test your design; turn in the VHDL entity and architecture, and a screen capture of a simulation trace showing the multiplexer's output for two distinct input words and both sel=0, sel=1.

Question 3.4: What is the worst-case propagation time from input to output of your 32-bit multiplexer?
Question 3.5: Explain the differences if any between the worst-case propagation times for your 3 multiplexer designs?

As you might expect, dealing with a 32-bit processor there will be situations in the design process where you will need to store a 32-bit number without storing it in memory. For this task you will need a 32-

bit register, "reg32", that on the rising edge of the clock signal, "Clk", latches a 32-bit input word, "D", if the write enable signal, "wr", is asserted; if "wr" is deasserted, the register's output word "Q" is unchanged. An asynchronous, active-low reset signal "clr" that will immediately cause the "D" output to go low also needs to be provided. Compile, simulate and test your design; turn in the VHDL entity and architecture, and a screen capture of a simulation trace showing the following sequence of events: the register is reset; the word "0x1234ABCD" gets written to the register in the first cycle after the reset and remains stored in the register for 10 cycles; the word "0xABCD1234" then gets written to the register and remains stored in the register for an additional 2 cycles. Below is block diagram that might help:



Question 3.5: What is the fastest clock rate that your 32-bit register can operate at correctly (remember that clock rate is not the same as worst-case propagation delay)?

As you have covered in class, the immediate field of the Immediate instruction is only 16-bits wide. If you want to use the immediate value of an Immediate instruction as the input to a component that only accepts 32-bit input signals you will need to extend the 16-bits to 32 either by zero or sign extension. In zero extension you simply concatenate 16 zeros with the input signal to form a 32-bit output signal with zeros as the 16 most significant bits. Sign extension is essentially the same thing except the 16 most significant bits of the output word take on the value of bit 15 in the input signal.

Design and simulate a 16-bit to 32-bit zero extender called "zeroext". Your input signal should be called "in0" and your output signal should be called "out0". Compile, simulate, and test your design; turn in the VHDL entity, architecture, and a screen capture of a simulation trace showing the zero-extender's output for input words "0x7FFF" and "0xFFFF".

Design and simulate a 16-bit to 32-bit sign extender called "signext". Your input signal should be called "in0" and your output signal should be called "out0". Compile, simulate, and test your design; turn in the VHDL entity, architecture, and a screen capture of a simulation trace showing the sign-extender's output for input words "0x7FFF" and "0xFFFF".

Combine your "zeroext" and "signext" components into a single component called "extender" which will perform a zero extension on "in0" when a new 1-bit input signal called "Sel" is equal to '0' and a sign extension when "Sel" is equal to '1'. Compile, simulate, and test your design; turn in the VHDL entity, architecture, and a screen capture of a simulation trace showing the sign-extender's output for input words "0x7FFF" and "0xFFFF" when "Sel" is both '0' and '1'.

Lab Report
Each of the remaining Assignments should follow the format discussed below:

1. The report should contain the following sections
   - Cover Page – with your name, date, and assignment number

- Introduction – briefly comment on the assignment and what it accomplishes
- Design and Testing:
    - Description of the component and its function. (If relevant, provide boolean description.)
    - Well commented VHDL code (if longer than 2 pages please attach in the appendix)
    - Tests that were performed to verify that component works correctly (if you provide waveforms make sure to describe the test they represent. In other Words ANOTATE!)
    - Answers to the questions in section 3.
- Textbook questions
- Appendix

2. The report needs to be typed (hand written notes are only allowed as comments on waveforms or any other graphics).
3. Any waveform that is not commented will not be considered as valid.
4. Make sure you use the names for inputs, outputs, and entities as outlined in the assignment. This will be essential for connecting these components together in later assignments.

Note: Since all reports will be submitted electronically, annotating your waveforms can be challenging and/or time consuming especially if you're going to use a program like Paint, Photoshop, or OneNote. I've found an open source screenshot and annotation utility called Greenshot http://greenshot.org that can make the process quick and easy.