

## Assignment 3 – Part I

**Due Date: 11:59 pm, Friday, June 22<sup>nd</sup>, 2007.**

### 1 Purpose:

This assignment will familiarize you with two aspects of computer architecture: i) the operation of a simple datapath from the perspective of hardware interpretation of instructions, and ii) the hardware - operating systems interface. Consequently the assignment has two parts. The first part is essentially a warm-up with a non-trivial implementation for which you will be given specific step-by-step instructions for extending the single cycle datapath. You will be given the VHDL model and will modify the same. In the second part you will be given a basic multi-cycle implementation of the same datapath. You will extend this datapath with instructions that are typically used by the operating system – for example to handle exceptions and synchronization. Part II will be posted in a few days. **Note that Part I and Part II are due at the same time!** However, in the interests of preparing for the first exam, it is highly recommended that you attempt to complete Part I by Wednesday June 13<sup>th</sup> as preparation for the first exam. As a practical matter, Part II will take more time.

### 2 Assignment

Part I has two components. The first component is a warm-up sequence of mechanical steps to help you become familiarized with the code. The second part requires you to update the datapath to implement a specific instruction that is not currently supported by the datapath.

#### 2.1 Warmup

While the single cycle model has been used in several prior classes and should be quite stable, please do report any discrepancies immediately.

1. To familiarize yourself with the model, determine the answer to the following. You may ask the TA and discuss this with your classmates and use the WebCT discussion group that will be monitored by the TA. There are no submission requirements for this component of Part I. It is to your benefit to complete this and understand the answers.
  - a. In which VHDL modules are the following components implemented?  
Note that this may not exactly correspond to what you understand from the figures in the text.
    1. The multiplexor controlled by the MemToReg control signal
    2. The logic for computing the branch address
    3. The multiplexor controlled by the RegDst control signal.
  - b. How many words of data memory are there?
  - c. What is the width (#bits) of the program counter?

2. Edit the data memory module to place the value 0x55 at location 0x4 (i.e., the second word in memory) and the value 0x66 at location 0x8 (the third word in memory). Remember this is a byte addressed machine.
3. Write a short program to load the word at location 0x4, the word at location 0x8, add the two values and store the result at location 0xc.
4. Assemble the preceding program. It is suggested that you use SPIM to assemble the program and verify it manually. Edit the fetch module (IFETCH.VHD) to store this program in by editing the instruction memory array to be have the hexadecimal values of the assembled instructions starting at location 0.
5. Compile and execute the single cycle data path. From the trace identify the following demonstrating to yourself that your program executes correctly
  - a. Encoded instructions
  - b. The operands as they show up at the output of the ALU
  - c. The results of the execution of each instruction as they show up at the input to the data memory.

## 2.2 Instruction Extension

Extend the single cycle datapath to implement the `lui` instruction. The following sequence of steps is a highly recommended solution approach although it is not required, i.e., you may follow a different sequence steps. **This component of the assignment must be done individually!** Note that the raw number of lines of VHDL code required is very little – on the order of a few lines. Most the effort will be in understanding what needs to be done (item 1 below) and debugging (VHDL & ModelSim).

1. Using a figure of the single cycle datapath, draw the modifications necessary for the datapath to correctly implement the instruction.
2. From the preceding determine the values of all (and any new) control signals required to realize the desired functionality.
3. Update the controller description to implement the new functionality.
4. Modify the individual VHDL modules to make the changes to the module.
5. Compile and test.
6. Capture the trace for submission.

## 3 Tools

When you use ModelSim, ensure that you add to the trace all signals in the top level of the model (MIPS.vhd). When you follow the ModelSim tutorial you will find a description of how to do this. If you have a problem see all of the signals, ask (in person or via email) the TA or me, or post to the WebCT discussion group. Do not spend too much time (you should spend some time to get up to speed and a requisite comfort level but not too much) on tool specific issues. I would rather you spend most of the time on the assignment.

You can use ModelSim simulation environment in the laboratory, or download the same as provided in the following instructions. It is highly recommended that you go through the ModelSim tutorial that is available from the **Help** button on the toolbar.

The following steps describe how you may install ModelSim on your personal machine.

1. You will have to Register an account with Xilinx first. Here is the link: <https://secure.xilinx.com/webreg/login.do?goto=https%3A%2F%2Fsecure.xilinx.com%3A443%2Fwebreg%2Fregister.do%3Fgroup%3Dmyprofile%26languageID%3D1>
2. Go To: <http://www.xilinx.com/ise/mxe3/license.htm>
3. Click "I Agree" at bottom of page
4. Click on "mxe\_3\_6.2c.zip"
5. Save the file. Open the zip file and double click on the "mxesetup.exe" application. You may be asked to extract more files, click "Extract All".
6. Then Click on "mxsetup.exe" from the new extracted folder.
7. Choose the Starter version (free).
8. Select "Full Vhdl" from the library installation option.
9. Run the Licensing Help. After you receive the license in an email, run the Licensing Wizard from the ModelSim folder in the Start menu.

You may use other VHDL tools rather than ModelSim. If you do, please ensure that your submission is in keeping with the detail required for the TA to grade.

## 4 Submissions Instructions

There is no submission requirement for the warm-up. For the instruction extension component, submit the following in electronic format.

1. A figure illustrating the modifications made to the single cycle datapath with all modifications clearly shown and all control signals clearly marked (scan this to a PDF).
2. A screen shot of the trace.
3. The complete VHDL code for the modified datapath.
4. Create a zip file of the preceding and email to the TA by the submission deadline.

## 5 Grading Guidelines

1. Clearly a complete and correct design as in items 1 & 2 of the submission: 35 points.
2. Compiles and executes the program (does not crash, appears to be correct): 30 points
3. Execution is correct: 25 points
4. Clarity of description and documentation (including code documentation): 10 points