Computer Systems Architecture I

Final Project Requirements and Report

This document indicates what is expected of the final project. It is divided into five parts. Portions of the various sections have been developed for PM1 and PM2 and can be reused as part of the overall documentation. Since improvements and modifications occur during the design process, changes in previous design components are expected. The final project and report will include the following components.

- **Design**: Describe the design of the processor. Indicate how the design deals with hazards, multithreading, and other items.

- **Design analysis**: Using performance equations from the text, analytically determine the performance that can be expected.

- **Debugging**: It should be possible to demonstrate the operation of each instruction, the execution of the two application programs, and the operation of multithreading.

- **Experiments**: Using your simulation, experiments should be performed and results reported.

- **Performance analysis**: The experimental results should be compared with the analytical performance models that you developed; the comparison should be discussed in your report.

- **VHDL implementation**: Describe the approach taken in the constructing your VHDL model.

- **Final Report and Demonstration**: The material above should be written up in a report. You will also be required to demonstrate the operation of your VHDL model.

Your report should be divided into (at least) six sections. The following sections describe the minimum structure required and describe in greater detail how the elements above should be reflected in your report.

1. **Processor Design**

   This section presents and documents the architecture of the five-stage, multithreaded pipelined processor.

1.1 **Basic Processor Elements**

   The section should contain the following elements

   1. **Instruction set**: Specification of the instruction set and an indication of instruction operation and addressing modes. This should include a brief discussion of the reasons for selecting this instruction set.

   2. **Instruction encoding**: Specification of the instruction encoding. The reasoning behind the encoding should be stated.

   3. **Registers and special features**: A listing of all register files, special registers, interstage registers, etc. should be present along with their size, addressing (wherever necessary),
and short description of their function. Your processor should operate in either single threaded or dual threaded modes (e.g., you should not need two threads for the processor to function).

4. **Data flow diagram**: A block diagram of the processor data flow should be present (similar to Figure A.18 in H&P). The diagram should be explained and the registers associated with the previous item identified.

5. **VHDL modules**: The separate VHDL modules associated with the design presented above should be presented and discussed.

### 1.2 Processor Pipeline Control Hazards

This section should contain the following elements:

1. **Hazards**: List all the data and control hazards that may be encountered with your pipelined design. Indicate how you deal with these hazards in your design. For each possible hazard, give an example of instruction sequences flowing through the pipeline. Indicate clearly the performance degradation due to the hazard.

2. **Pipeline stage events**: A listing of events associated with each pipeline stage similar to Figure A.19 in H&P should be present and explained.

3. **Multithreading control**: Indicate how your processor operates in and transitions between single-threaded and dual-threaded modes.

4. **Pipeline control**: The control mechanism used for the pipeline should be presented and explained. For example, if a global processor state has been defined and control is exercised by use of a state transition diagram, then the diagram should be presented and the control signals that initiate events should be specified. It is important that a coherent and understandable control scheme is used.

5. **VHDL implementation**: A discussion of the VHDL implementation of the control scheme described above should be presented. This part of the report is introductory in nature, so the section should give an indication of how the processor's pieces have been mapped onto modules. This is good place to: a) include the VHDL flowchart you created, b) described it briefly, c) call attention to anything you consider unique or interesting in your VHDL implementation, and d) refer to Section 6 for the code and brief description of each of the majors modules.

### 2. Design Analysis

This section should contain the elements below. (For parts 1-5, feel free to consider single-threaded mode only.)

1. **CPI estimation**: Using the statistics associated with SPECint (e.g., see page B-41 of H&P), estimate the ideal CPI for your machine.

2. **CPI for two applications**: Using the two programs that you developed, estimate the ideal CPI for those programs when executing on your machines. Be sure to state all assumptions needed to make your estimate.

3. **Memory accesses/Instruction**: For your design and the two programs, determine the memory access to instruction ratio. How does that compare with SPECint?
4. **Instruction count**: For your design and the two programs, determine the instruction count (i.e., the number of dynamic instructions executed).

5. **Execution time estimates**: For the two programs you have developed, estimate the execution time using the analytic formulas given in the text. Obtain expressions for these time estimates as a function of the miss rate and miss penalty.

6. **Multithreading**: Develop an approximate expression for the throughput of the processor (instructions per clock) for your processor operating in single threaded and then dual threaded mode.

7. **Stage execution time balance**: Do an approximate analysis of the time associated with each stage in the pipeline to determine how well the stages are balanced. Assume that simple register operations (loading, reading) take 1ns, simple arithmetic and logical operations take 2ns, multiply takes 6ns and divide takes 12ns. Say that complex logical operations (used in control tasks) take 1ns * (the number of levels of logic that are needed to execute the function). Assume that memory reads take 4ns. Estimate the number of logic levels as best you can. To do this, estimate the approximate logic gate depth of each pipe stage; in other words, how many stages of logic would be needed to implement the critical path? A 2-input "and" needs one level, for example.

8. **Processor MIPS rate**: estimated the processor MIPS rate based on your pipeline execution time estimates (e.g., base your hypothetical clock on the slowest pipeline stage).

**3. Debugging**

Develop a methodology for debugging your processor design. The objective here is to ensure **correct** processor operation. Later we focus on performance issues. This should include, at a minimum, the following.

1. **Single instruction execution**: A procedure should be available for demonstrating that each of the instructions in each of the addressing modes operates correctly.

2. **Simple programs for single threaded operation**: Several simple programs containing just a few instructions should be tried to ensure correct operation in single threaded mode.

3. **Simple programs, single threaded, hazard operation**: Several simple programs containing just a few instructions should be tried to ensure correct operation in single threaded mode when hazards are present.

4. **Simple, programs, dual threaded operation**: The simple programs use for the prior two situations should also be used to debug multithreaded processor operation.

5. **Sort and Stats programs**: The sort and statistics programs should be tried under single and dual threaded operation. It should be possible to change the data and observe program operation under different input data sets.

6. **VHDL Model verification**: Include several VHDL timing diagrams and annotate the diagrams to indicate key actions (hazard detection, context switches, stalls, etc.).

7. **Other tests**: Include any other tests required to demonstrate proper operation of your processor.
4. **Experiments**
For each of the elements in the Design Analysis section above, there should be an experiment that obtains execution time data for the various performance parameters. For example, experimentally determine the CPI for each application, as well as the memory access per instruction, the execution time and the instruction count. Run a set of experiments to determine the effects of multithreading on processor throughput. Vary the miss rates and miss penalty and develop an experimental set of graphs that illustrate processor performance as a function of miss rate, miss penalty, number of threads, etc.

5. **Performance Analysis**
The experimental results obtained in the prior section are to be compared with the analytical performance models developed in the Design Analysis section. The differences between the two should be discussed, and, where possible, explained. The general implications for processor design associated with the results should be considered.

6. **VHDL Implementation**
The approach taken in the VHDL model development and coding should be described and discussed. List and briefly describe the main modules and their associate with the processor description. If a hierarchical approach was taken, indicate the structure of the hierarchy. Include your code in an appendix and make sure your code is well commented.

7. **Other Report Details**
Your report should not exceed 25 pages (double-spaced, 11 point font size, one inch margins) excluding diagrams, graphs, and VHDL code. The final report will be due at the time of your project demonstration. You will need to sign-up for a 30-minute demonstration slot. Available dates and times will be posted soon (they will occur the week before and during finals).