20-EECE-681—Quarter Project--Week 2
April 7, 2009
Due Monday April 13

Project modifications:

1. we have changed the language for your project to Verilog. Verilog references are available on the syllabus and also on the blackboard site, under "document".

2. you are allowed to use the Synopsis cvs tool available on the Sun workstations in 890 Rhodes to compile your Verilog code and also to run testbenches. You will still need to use the Altera tools to obtain physical information such as timing and power usage, but you can use testbenches in the synopsis tool for functional verification.

3. sample code and information for the benchmark programs (not tested for accuracy):

Problem 1. mergesort:
http://linux.wku.edu/~lamonml/algor/sort/merge.html

Problem 2. quicksort:
http://linux.wku.edu/~lamonml/algor/sort/quick.html

NOTE: the following two problems are significantly more difficult than the first two, in terms of the optimizations that can be performed.

Problem 3. matrix multiplication (integer entries):
There are many techniques for optimizing matrix multiplication. You need to choose carefully which one(s) to try. Note that the matrix size is important. You may concentrate on square matrices.
comparison of several methods:
http://wwwcsif.cs.ucdavis.edu/~koike/ecs289k/projecta/
interesting comments on matrix multiplications:
http://www.nersc.gov/~simon/cs267/hw1note/optimization.pdf
http://surj.stanford.edu/2004/pdfs/kakaradov.pdf
Wikipedia article:
http://en.wikipedia.org/wiki/Matrix_multiplication
Chapter 28 of Cormen, Leiserson, Rivest, and Stein also deals with matrix operations, including matrix multiplication.

Problem 4. FFT:
Here is a simple explanation from Dr. Dobbs:
http://www.ddj.com/cpp/199500857
This is also covered in Chapter 30 of Cormen, Leiserson, Rivest, and Stein.

Basic Architecture:
The basic architecture for your project will be the LC-3 architecture defined in the book by Patt and Patel. Your basic processor should be able to perform all the operations of this processor. Information on this processor is available in the "Student Resources" section of the book web page at

http://highered.mcgraw-hill.com/sites/0072467509/student_view0/appendices_a__b__c__d____e.html

The instruction set is given in Appendix A and schematics and information on control are given in Appendix C. There is also a downloadable simulator.

Note that you will probably want a multiply instruction for benchmark problems 3 and 4. It is up to your group to decide how to add this instruction to the architecture. Two strategies we discussed in class were:
i. replace one instruction
ii. make the op code 5 bits long.
You may think of others.

Week 2 tasks:

1. Modify your week 1 assignment if necessary according to the comments you received.

2. CARRY OUT the experiments we designed in lecture for the simple problems on critical path, delay, and fanout (and power).

2. Begin work on the datapath of your processor.

Step 1. Explain completely and justify any changes you will be making to the ISA (instruction set architecture) of your machine. These should be related to operations needed by the benchmark problems or optimizations you plan to carry out in steps 2 (pipelining) and 3 (dual processors). Explain any changes to the capability of the LC-3 processor which your modifications will make and justify these.

Step 2. Design and test the datapath of your processor. This step will take two weeks, so the report your submit next week should have the toplevel design requirements and test and as much of the rest of the design as you have completed. The rest will be done for Project Report 3.

Step 3. Complete and submit your weekly report, with updated Gantt and tracking charts.

Steps to be used in design and testing for your architecture for this and subsequent reports:

1. Design Phase

a) Create the requirements specification for each task in the project assignment. Ensure the requirements specification is a) complete, b) decomposed to appropriate levels to support the design function, and c) correctly specifies the functionality, performance, and implementation resources of the project.
b) Design and verify (using Verilog models) the digital circuits required to implement the project such that the project requirements and your requirement specifications are fully and correctly implemented.
c) Design a set of tests, both test models written in Verilog and the data or data generators necessary to conduct the tests. Validate each test and test input data by executing the Verilog models developed in 1.b. The purpose of the test set is to validate that the implemented circuit meets the requirements specification.
d) Using the Verilog circuit, testbenches, and the Altera tools, answer the questions given in the Performance Data and Results section below. The performance data thus obtained is what you expect to get when you implement the circuit.
e) Document the requirements specification, circuit design, test set design, and expected performance in your report.

2. Implementation / Evaluation Phase.

f) Create a single integrated test set.

g) Synthesize, configure, and execute the design on the Altera board.

h) Apply the integrated test set to fully verify correct execution of the circuit, and to obtain performance data

3. Performance Data and Results

The following questions are to be answered for both the simulation and the actual circuit

1. What is the minimum execution time (to an accuracy of 1 μs) of your implemented circuit for each benchmark? Minimum execution the time to execute a benchmark assuming 1) all data required to execute is in FPGA memory, and 2) the frequency of the FPGA clock is maximum for correct operation of the benchmark.

2. What FPGA area resources (columns, CPB's IOB's, cells, or any other significant resource units) does your circuit use?

3. How much memory (in units of bytes) is used to store the machine code and associated data for each benchmark?

4. What is the power usage?