10220 CS410001-Computer Architecture 2014

Project #1

1. Project Objective
   a. Implement a single-cycle, functional processor simulator according to the reduced MIPS R3000 ISA specified in *Appendix A*, "*Datasheet for the Reduced MIPS R3000 ISA*."
   b. Design your own test case to test the functionality of your simulator.

2. About the Simulation
   a. The execution of the single-cycle processor simulator should **terminate** after **executing the "halt"** instruction.
   b. Both the instruction memory and data memory are in **little-endian** format.
   c. **Register 0** is a **hard-wired constant 0**; any attempt to write to register 0 takes no effect.
   d. Assume that both the instruction memory and data memory are of 1K bytes size.
   e. You should implement an **error handler** to deal with erroneous instruction executions. For more details please refer to *Appendix D*, "*Error Detection Samples*."
   f. The simulator should get input files and generate output files at the same directory as that of the executable. Note that the executable file should be named **single_cycle** and takes no command-line arguments.

3. Input Format
   a. For each test case, **all registers, except PC and $sp, are initialized to 0's**; other initial conditions are specified in the following two files.
      i. **iimage.bin**: The instruction image (little-endian format, encoded in binary). The first four bytes indicate the initial value of PC, **i.e. the starting address to load the instruction image**. The next four bytes specify the number of words to be loaded into instruction memory (I memory). The remaining are the program text to be loaded into I memory. All other addresses not covered by the image are assumed to have been initialized to 0's.
      ii. **dimage.bin**: The data image (little-endian format, encoded in binary). The first four bytes show the initial value of $sp. The next four bytes indicate how many words to be loaded into data memory (D memory). The remaining are the data to be loaded into D memory, starting from address 0. The contents of all other addresses not covered by the image are assumed to have been initialized to 0's.
   b. For details about input format, please refer to *Appendix B*, "*Input Samples*."

4. Output Requirement
   For each test case, please respectively output to the following two files:
   a. **snapshot.rpt :** contain all register values at each cycle.

b. **error_dump.rpt :** contain error messages.

For more details please refer to *Appendix C-1*, "*Output Samples for Project 1"* and *Appendix D, "Error Detection Sample*".

5. Test case design
   a. Your test case should be of the same format as described in list 3 and *Appendix B*, and both "**iimage.bin**", "**dimage.bin**" should be placed under a folder named **testcase**.
   b. Note that the initial value of PC is also the starting point of the execution of your program specified in **iimage.bin**.
   c. Your test case should run no more 500K cycles.

6. Submission
   a. Submit a folder named ***archiXX,*** where XX is your group ID, and under the folder provide the following **two folders** and **PDF file**:
      i. ***simulator/*** : contains your「**Makefile**」,「**README**」, and source codes.
      ii. ***testcase/*** : contains your test case files.
      iii. ***archiXX_report.pdf,*** where XX is your group ID
      - Your **Makefile** should support two functions:
        make – Build your simulation environment
        make clean - Erase from the build tree the files built by make all.
   b. Finally, compress the folder ***archiXX*** as ***archiXX.tar.gz***, and upload *archiXX.tar.gz* to the iLMS system.
      For example, if you are of group 0, then you should upload *archi00.tar.gz* to iLMS system.

- **Note 1**: each project has two due dates, and the second due date is one week after the first one.
- **Note 2**: At your first submission, the TA team will not use your test case for grading, but will generate reference output files (**snapshot.rpt, error_dump.rpt)** for you using TA's golden simulator. You may use the reference files to correct your test case or simulator.
- **Note 3**: For the second submission, TAs will evaluate your simulator, test case, and report file.

7. Evaluation
   a. Correctness of simulator: 70%
      i. Open test cases : 30%
      ii. Hidden test cases: 40%
      - **Note 1**: we will release the open test cases before the first due date, and you may use

them to verify your simulator.

- **Note 2**: for the 2nd submission all TA's test cases will become open, and students' test cases will become the hidden test cases.
- **Note 2**: We will take the average score of the two submissions as your final simulator score.

b. Report: 10%
   i. Report is limited to 10 pages.
   ii. Your report can be either in Chinese or English, or mixed. The report should be named archiXX_report.pdf, where XX is your group ID.
   iii. Grading principle:
      1. Simulator design (2.5%): Discuss if any special data structures, design patterns, code structure, and general execution flow, etc.
      2. Simulator elaboration (2.5%): Clearly explain your design with assistance of tables, figures, etc.
      3. Test case design (2.5%): How you implement your test case in C code and assembly code? What corner cases are you targeting at? We will also look for your creative ideas used in your test case.
      4. Test case elaboration (2.5%): Comments to both C code and assembly code of your test case; the mapping between variables and register/memory locations and that between codes and labels; etc.

c. Test case: 20% * (1 – $[1.5]^{-n}$), n: number of defeated groups
   i. A test case is valid if the output generated by TA's simulator is the same as that from your own simulator.
   ii. **You get zero points if your test case is invalid.**
   iii. The score of your test case is evaluated based on the second submission.

Note that we use **nthucad workstation** as our official testing environment.
Furthermore, we will evaluate your project using scripts. It is your responsibility to make sure that your project can be executed by the script provided by TAs. **If it cannot run through the official script, you will get zero points even if your program or result is correct.**

8. Etiquette

a. Do not copy others work, or you will fail this course.
b. No acceptance of late homework.
c. **Please frequently check the class website announcements for possible updates.**