10220 CS410001 – Computer Architecture 2014

Project #2

1. Project Objective
   a. Implement a pipelined, functional processor simulator for the reduced MIPS R3000 ISA, following the specification "*Datasheet for the Reduced MIPS R3000 ISA*" in *Appendix A*
   b. Design your own test case to validate your implementation, particularly on hazards handling.

2. Additional Specification
   a. Pipeline Description
      i. 5 stages: instruction fetch (IF), instruction decode (ID), ALU execution (EX), data memory access (DM) and write back (WB).
      ii. Conditional branches and unconditional branches are evaluated during the ID stage.
      iii. Load/store computes the address to be accessed in D memory during the EX stage, and accesses data memory during the DM stage.
      iv. Arithmetic/bitwise shift/logical operations are done during the EX stage.
      v. There are three forwarding paths: EX/DM to ID, EX/DM to EX, DM/WB to EX.
      vi. All **write back** executions targeting to \$0~\$31 **are done during the first half of the cycle** in the WB stage.
      vii. All **reads to registers are done during the second half of the cycle** in the ID stage.
      viii. PC is updated in each cycle **after** executing all instructions in each pipeline stage.
      ix. If inserting "NOPs" is needed to resolve hazards, use *sll \$0, \$0, 0*, i.e. the bit stream $0x00000000_h$. **In your implementation, you should decode** the instruction bit stream $0x00000000_h$ **as NOP**.
   b. Other Constraints
      i. The pipeline is **initialized** with **NOPs in all stages**.
      ii. The simulation of the pipelined processor **terminates** after the **first "halt"** instruction arriving **at the stage WB**.
      iii. **Register 0** is a **hard-wired 0**; any attempt to write to register 0 takes no effect.
      iv. The instruction memory is of 1K size, the data memory of 1K size.
      v. The executable should be named **pipeline**.
      vi. To avoid re-execution of some stages, the order of simulating the pipeline is WB → DM → EX → ID → IF.
3. Input Format
   Same as that of Project 1. Please refer to the specification of Project 1 and *Appendix B*, "*Sample Input*."

4. Output Requirement

For each test case, an output file named **snapshot.rpt** and error_dump.rpt should be generated.

a. **snapshot.rpt :**The file should contain the values of all registers, mnemonic in each pipeline stage and hazards at each cycle.
b. **error_dump.rpt :** The file should contain error messages.

For format details please refer to *Appendix C-2*, "*Sample Output for Project 2*." and *Appendix D-2*, *"Error Detection Sample for Project 2*".

5. Test Case Design

Your test case should cover at least one control hazard, one data hazard resolved by inserting NOPs, and one data hazard cleared by forwarding. The number of NOPs inserted is not limited, and same is the forwarding path used in your test case. For other details, refer to *project_1.pdf*. **Note that your test case should not run over 500,000 cycles, or it will be deemed invalid.**

6. Evaluation

Same as project 1.

Note that all evaluations, verifications will be done on **workstation (140.114.75.140)**. Furthermore, we will evaluate your project using scripts. Please make sure that your project can be executed by the script provided by TA's. **If it cannot run through the script, you will lose all your points even if your program or result is correct.**

7. **Etiquette**
   a. **Do not copy others' works, or you will fail this course.**
   b. **No acceptance of late homework.**
   c. **For details of submission, please note the announcement on the course website.**