Computer Design and Organization
Assignment #1

Due: Wednesday October 9

This assignment has two parts: (1) a "paper and pencil" exercise that has to be done individually and (2) a Bliss experiment that you should do in teams of two.

Part I

The 5 stage MIPS pipeline (IF,ID,EX,Mem,WB) is not the only way to organize a 5 stage pipeline. In the following we will refer to this organization as the LUI pipeline where LUI stands for Load-Use Interlock.

A second organization, used in the first Pentium consists of the 5 stages (IF,ID,AD,EX/Mem,WB) where IF, ID, and WB are as in the LUI pipeline but the AD stage calculates the effective address of a memory operation and the EX/Mem stage either performs an arithmetic-logical operation as the EX stage of LUI or accesses memory as the Mem stage of LUI. This second organization is called AGI for Address Generation Interlock.

In both LUI and AGI, branches are resolved in the first half-cycle of the EX stage, i.e., in the absence of branch prediction one "bubble" must be inserted after a branch in the LUI case and two bubbles in the AGI case.

Discuss the advantages and drawbacks of each organization with respect to data and control hazards and hardware (functional units, forwarding paths) requirements. Give MIPS-like sequences of instructions illustrating these advantages and drawbacks.

Part II

The purpose of this first assignment is to acquaint you with the *Bliss* environment. Refer to the CSE471 homepage on how to run *Bliss*. **Look at the *Quickstart* link to see how to set-up this first assignment.**

In order to get an idea of warm-up effects on simulation results perform the following experiments using the SPEC benchmark *perl* (executable is perlbmk_base.oct6a, input is recurse.t, see *Quickstart*).

- Experiment 1: gather statistics for the first 1 million instructions
- Experiment 2: skip the first 5 million instructions and gather statistics for the next 1 million instructions.
- Experiment 3: skip the first 4 million instructions, then warm up the simulator for 1 million instructions and then gather statistics for the next 1 million instructions.
- Experiment 4: skip the first 50 million instructions and then gather statistics for the next 1 million instructions.
- Experiment 5: skip the first 49 million instructions and then warm up the simulator for 1 million instructions and then gather statistics for the next 1 million instructions.

In order to perform these experiments you must change the following variables in the standard configuration files:

- -System.FastForwardAmount :number of instructions skipped
- -System.WarmUpAmount :number of warm-up instructions
- -System.SimulateAmount :number of instructions simulated

NOTE 1: System.WarmUpAmount is counted in System.SimulateAmount. So, for example if I wanted to skip 50000 instructions, warm-up for 40000, and simulate for 100000, I would set up:

- -System.FastForwardAmount :50000
- -System.WarmUpAmount :40000
- -System.SimulateAmount :140000

NOTE 2: If you don't set-up System.SimulateAmount, the program will run to completion, i.e., it will take about 1 hour to simulate. Beware!

NOTE 3 You might also want to rename appropriately the output file where statistics are recorded.

Answer the following questions in a format similar to the one posted on the Web page (cf. *Report guideline*).

1. Which of the five experiments do you think is most likely to be representative of the performance of the perl benchmark? Explain your choice.
2. The standard configuration file indicates that 4 instructions can be in flight at the same time (i.e., executing in the broad sense). Why is the IPC so far from 4?
3. Why is the number of L1DataCache.WriteBacks lower than the number of L1DataCache.Misses?
4. Why is the L2Cache.HitRatio so much smaller than the L1DataCache.HitRatio?