LSU EE 4720 Homework 3 Due: 9 April 2008

For answers to these questions consult the SPECcpu2006 Run and Reporting Rules (which can be found at `spec.org`*).*

Problem 1: One way testers can stretch the rules is by using compiler optimizations that give good performance when they work correctly but are too error prone for non-experimental use.

(*a*) Why would it be a bad idea for SPEC to limit allowable compiler optimizations to those that are already known to be safe? (Say, dead-code elimination based on a SPEC-provided analysis technique.)

(*b*) Rather than dictate allowable optimizations the rules instead explain that if it's good enough for your customers it's good enough for SPEC, though not in those words. Find the section in the run and reporting rules where this rule is given.

(*c*) For at least three bullet items in the section (from the last part) explain what sort of unscrupulous action the bullet item is supposed to prevent.

Problem 2: When preparing a run of the SPEC benchmark the tester provides, among other things, libraries (such as the C standard library that contains routines such as strlen, malloc, printf). It is in the testers interest to make sure these library routines run as fast as possible and is free to do so within the SPEC rules.

Section 2.1.2 stipulates that one can't use flags that substitute library routines for routines *defined in the benchmark*.

In addition to base and peak, imagine a third metric called *swap*, in which the rule in Section 2.1.2 didn't apply. Testers could abuse the swap metric by substituting routines that merely return the correct value (since input data is known in advance), but for this question suppose testers comply with the spirit of the SPEC rules and substitute routines which provide higher performance for any input data.

(*a*) Comparing the peak scores to the base scores shows the additional performance that can be obtained by a suitably motivated and resourced expert. Explain what might be learned by comparing swap scores to base and peak scores. (That is, where might the higher performance be coming from.)

(*b*) Provide an argument that the swap metric is a good test of a system that complements base and peak.

(*c*) Provide an argument that swap doesn't really tell you anything about the system (CPU, memory, compiler and other build items).

Problem 3: For exceptions the handler needs to know the address of the faulting instruction both so that it can examine the instruction and so that it knows where to return to in case the instruction needs to be re-executed or skipped. *For answers to this question consult the ARM and MIPS32 (Volume 3) ISA manuals on the course references page.*

A programmer-friendly ISA would provide the handler with the address of the faulting instruction, however in both MIPS32 and ARM may provide an address *near* the faulting instruction.

(*a*) In which registers do MIPS and ARM A32 write the approximate faulting instruction address? (For MIPS give the register number as well as its name.)

(*b*) The address that MIPS provides may be that of the faulting instruction, or it may not be. When is this done, and what is the other address?

(*c*) ARM A32 also does not provide consistent addresses. What addresses does it provide? Give a credible reason for the differences in addresses.