6.823: Lab 1 Questions

Nada Amin
`namin@mit.edu`

Due: 29 September 2008

1. Assuming both architectures resolve all hazards, I would choose Architecture A, because the forwarding path would avoid bubbles and thus increase the instruction throughput.

   However, I am uncertain that Architecture A can avoid all control hazards assuming the instruction set has undelayed branches. Indeed, if the condition of a conditional branch is only known at the end of the execute phase, it might be necessary to have a path from the execute stage to the fetch state in order to cancel the instructions being fetched (which might not match the branch condition).

   In addition, Architecture B might have less control logic and thus be cheaper to implement. Thus, it might be more suitable for a low-end system.

2. The ebp register accounts for most of the weight in the tail. The ebp register is written when entering a functional call and read when exiting. Functions have varying lengths tailing off, so the dependencies for ebp follow this pattern. We could get rid of these ebp-related dependencies by keeping a stack with all the base pointers for each stack frame.

3. By couting read/write to partial registers as read/write to their respective full registers, we are being over-cautious and skewing our histogram towards finding more and closer dependencies than there actually are.

   First, I think the tradeoff is in the right direction. It's better to overestimate dependencies than to underestimate them, because, this way, our estimates will be upper bounds and thus provide us some guarantees.

   Second, I ran the tests by counting read/write to partial registers separately, and the results for a dependency of 1 are off by less than 0.02%. Thus, it doesn't seem worth the effort to be more accurate in this respect.

4. I would guess that Architecture A has more registers because it has fewer 1-dependencies, which are the most troublesome.