ECE571: Advanced Microprocessor Design – Homework 3

Due: Friday 10 October 2014, 5:00PM

Create a document that contains the data and answers described in the sections below. A .pdf or .txt file is preferred but I can accept MS Office or Libreoffice format if necessary.

1. **Branch ratios on x86 Haswell Machine**

   For this section, log into the Haswell machine just like in HW1 and HW2.

   In class we said traditionally computer architects say that one in five instructions is a branch. Here we run some experiments to see if that is true.

   (a) Run the bzip2 benchmark and measure `instructions:u`, `branches:u`, and `r5301c4:u` (`r5301c4:u`, according to libpfm4, corresponds to the `BR_INST_RETIRED:COND` event which measures only conditional branches).

   ```
   perf stat -e instructions:u,branches:u,r5301c4:u
   /opt/ece571/401.bzip2/bzip2 -k -f ./input.source
   ```

   What are the total number of instructions, branches, and conditional branches? What is the ratio of branches to total instructions? What is the ratio of conditional branches to total instructions?

   (b) Now do the same test with the equake_l benchmark.

   ```
   perf stat -e instructions:u,branches:u,r5301c4:u
   /opt/ece571/equake_l.specomp/equake_l <
   /opt/ece571/equake_l.specomp/inp.in
   ```

   Report the total number of instructions, branches, and conditional branches as well as the ratio of branches to total instructions and ratio of conditional branches to total instructions.

2. **Branch miss rate on x86 Haswell Machine**

   (a) For the bzip2 benchmark measure `branches:u` and `branch-misses:u`. Calculate the branch miss rate (hint, you'll notice perf does this for you when you measure these instructions).

   (b) Also calculate the branch miss rate for equake_l.

3. **Speculative execution on x86 Haswell Machine**

   We want to measure the number of executed instructions vs the number of retired instructions. Haswell, unlike other processors, has no "executed instructions" event but instead we can use the μop events. libpfm4 tells us that `UOPS_RETIRED` is `r5301c2:u` and `UOPS_EXECUTED` is `r5302b1:u`

   (a) Find out what percentage of instructions were executed but not retired with bzip2.

   (b) Find out what percentage of instructions were executed but not retired with equake_l.

4. **Branch ratios on an ARM Embedded Machine**

   For this section you will log into an ARM Cortex A9 NVIDIA Tegra board. To do this, when logged into the Haswell machine run:

   ```
   ssh trimslice
   ```

   Your password should be the same as it is on the Haswell machine.

   Gather the results using perf. Note that the Cortex A9 does not support the :u "user only" postfix so your measurements will include kernel counts as well.

(a) Run the bzip2 benchmark and measure `instructions` and `branches`. (Remember to first copy the input file to your local directory).

```
cp /opt/ece571/401.bzip2/input.source .
perf stat -e instructions,branches
/opt/ece571/401.bzip2/bzip2 -k -f ./input.source
```

What are the total number of instructions and branches? What is the ratio of branches to total instructions?

5. **Branch miss rate on ARM Embedded Machine**

(a) For the bzip2 benchmark measure `branches` and `branch-misses`. Calculate the branch miss rate (hint, you'll notice perf does this for you when you measure these instructions).

6. **Short Answer Questions**

(a) Does the branch to instruction ratio differ between equake and bzip2? Why might this be?

(b) Does the branch to instruction ratio differ between bzip2 on the Haswell machine and bzip2 on the ARM machine? Why might this be?

(c) How do the branch miss-prediction rates compare between bzip2 and equake on the Haswell machine? What might be the source of any differences?

(d) How do the branch miss-prediction rates compare between bzip2 on Haswell and bzip2 on the ARM machine? What different design decisions might have been made between the two machines that affects these results?

(e) How do the executed vs retired instruction rates differ between bzip2 and equake on the Haswell machine? What implications might this have about the power efficiency of the two benchmarks?

(f) Imagine you wanted to write a benchmark to validate the branch prediction performance counters on a system. What kind of short benchmark could you write that would give you a 50% miss-predict rate?

Optionally write such a small example program, test it out, and report your results.

7. Submitting your work.

- Create the document containing the data as well as answers to the questions asked.
- Please make sure your name appears in the document.
- e-mail the file to me by the homework deadline.