LSU EE 4720 Homework 5 Due: 3 December 2002

To answer the questions below you need to use the PSE dataset viewer program. PSE (pronounced see) runs on Solaris and Linux; you can use the computer accounts distributed in class to run it, a Linux distribution may also be provided for running it on other systems.

Procedures for setting up the class account and using PSE are at `http://www.ece.lsu.edu/ee4720/proc.html`; preliminary documentation for PSE is at `http://www.ece.lsu.edu/ee4720/pse.pdf`.

Problem 1: Near the beginning of the semester the performance of a program to compute π was evaluated with and without optimization. It's back, down below.

Follow instructions referred to above to view the execution of the optimized and unoptimized versions of the pi program running on a simulated 4-way dynamically scheduled superscalar machine with a 48-instruction reorder buffer. The datasets to use are `pi_opt.ds` and `pi_noopt.ds`.

(*a*) Based on the pipeline execution diagram compute the CPI of the main loop for a large number of iterations in the optimized version. Do not use the IPC displayed by PSE, instead base it on the PED. In your answer describe how the CPI was determined.

(*b*) Consider first the optimized version of the program. Would it run faster with a larger reorder buffer? Would it run faster on an 8-way superscalar machine? How else might the processor be modified to improve performance? Explain each answer.

(*c*) Now consider the un-optimized version. Would *it* run faster with a larger reorder buffer? Should a computer designer pay attention to the performance of un-optimized code? Explain each answer.

(*d*) The simulated processors use a gshare branch predictor. Use `pi_opt.ds` to answer this question. How many bits is the global history register? Entries in the PHT are initialized to 1 and the GHR is initialized to zero. The PHT is updated when the branches resolve (in the cycle after they execute). Explain your answer.

(*e*) Would execution be any different if the PHT were updated when the instructions commit?

```
#include <stdio.h>

int
main(int argv, char **argc)
{
  double i;
  double sum = 0;                                                    // Line 7

  for(i=1; i<5000;)                                                  // Line 9
    {
      sum = sum + 4.0 / i;   i += 2;                                 // Line 11
      sum = sum - 4.0 / i;   i += 2;                                 // Line 12
    }

  printf("After %d iterations sum = %.8f\n", (int)(i-1)/2, sum); // Line 15

  return 0;
}
```