EN164: Design of Computing Systems **NAME______________**

Homework Assignment #4

Due **Friday, May 4, 2007, in class**

1) Suppose we have a deeply pipelined processor, for which we implement a branch-target buffer for the conditional branches only. Assume that the misprediction penalty is always 4 cycles and the buffer miss penalty is always 3 cycles. Assume also a 90% hit rate and 90% accuracy, and 15% branch frequency. How much faster is the processor with the branch-target buffer versus a processor that has a fixed 2-cycle branch penalty? Show your work and describe your reasoning. Assume a base CPI without branch stalls of 1.

2) Consider the following 2-level predictor. The first level is a 1024-entry direct mapped buffer, where each entry holds the history of the last 10 dynamic branches encountered for a particular (set of) branches. The second level consists of a table of 1K entries, where each entry holds a 2-bit saturating counter that provides the prediction. Using the nomenclature introduced in class, this is a PAg 2-level predictor (see slides 3-6 from lecture #28 on April 11 for related examples).
   a) Draw a figure that describes how the 1st and 2nd levels are organized and how they are indexed.
   b) How many bits total are required to implement this scheme?

.

3) To keep the issue step of a statically scheduled superscalar processor simple, an issue limit of one integer and one floating-point instruction per clock cycle may be imposed. Now, let's remove this restriction and see how the workload for the issue step grows with increasing multiple-issue capability. Assume a 5-stage superscalar pipeline (IF, ID, EX, MEM, WB) with no issue restrictions and no structural hazards. Also, regardless of instruction, each stage always takes just 1 clock cycle to complete its task, and an instruction may have up to two operands and one result.
   a) List the type(s) of data dependencies that must be checked for in the ID stage.
   b) For a 2-issue design with 32 integer registers and 32 FP registers, how many bits must be brought to comparators in the ID stage and how many comparisons must be performed during each clock cycle to check just for data hazards? How many if the issue width is doubled?
   c) Let the issue limit be *n* instructions, and assume the total number of registers is unbounded. How many comparisons, as a function of *n*, must be performed to check just for data hazards?

4) In small bus-based multiprocessors, write-through caches are sometimes used. One reason is that a write-through cache has a slightly simpler coherence protocol. Show how the basic snooping cache coherence protocol of Figure 4.7 on page 215 of the textbook can be changed for a write-through cache. From the viewpoint of an implementor, what is the major hardware functionality that is not needed with a write-through cache compared with a write-back cache?

5) Add a clean exclusive state to the basic snooping cache coherence protocol (Figure 4.7, page 215). Show the protocol in the same format of this figure.

6) Prefetching is a technique that allows the "consumer" of data to request data to its cache before they are needed. With multiprocessors, we may consider a similar technique where the "producer" of the data "evicts" the data from the cache after it is done with them. An extension of this eviction technique could be to send the data to the next processor that needs the data, in cases where that can be determined.
   a) When is prefetching likely to be applicable?
   b) When is producer-initiated communication likely to be beneficial?
   c) Would producer-initiated communication be applicable in the context of queuing locks?

7) Consider an SMT processor that has the capability of executing up to 8 threads simultaneously. Results collected from running a suite of benchmarks reveal that the branch mispredition rate nearly doubles from 5% to 9.1% going from 1 thread to 8 threads in the processor. However, the wrong-path instructions fetched (on a misprediction) drops from 24% on a single threaded processor to 7% on an 8-thread processor.
   a) What causes the increase in branch misprediction rate?
   b) Why is there a decrease in the number of wrong-path instructions fetched even though there is an increase in branch misprediction rates?
   c) Based on these numbers, what conclusions can you draw about the conflicts caused due to contention and interference on various resources in a multithreaded processor?