COMPUTER SCIENCES DEPARTMENT
UNIVERSITY OF WISCONSIN—MADISON
PH.D. QUALIFYING EXAMINATION

Computer Architecture
Qualifying Examination

Fall 2012

GENERAL INSTRUCTIONS:

1. Answer each question in a separate book.
2. Indicate on the cover of *each* book the area of the exam, your code number, and the question answered in that book. On *one* of your books list the numbers of *all* the questions answered. *Do not write your name on any answer book.*
3. Return all answer books in the folder provided. Additional answer books are available if needed.

SPECIFIC INSTRUCTIONS:

Answer all of the following **SIX** questions. The questions are quite specific. If, however, some confusion should arise, be sure to state all your assumptions explicitly.

POLICY ON MISPRINTS AND AMBIGUITIES:

The Exam Committee tries to proofread the exam as carefully as possible. Nevertheless, the exam sometimes contains misprints and ambiguities. If you are convinced a problem has been stated incorrectly, mention this to the proctor. If necessary, the proctor can contact a representative of the area to resolve problems during the *first hour* of the exam. In any case, you should indicate your interpretation of the problem in your written answer. Your interpretation should be such that the problem is non-trivial.

1. Translation Lookaside Buffers (TLBs)

Translation lookaside buffers (TLBs) are widely used to speed up the process of translation from virtual to physical addresses. Consider an architecture with 64-bit virtual addresses <$V_{63}V_{62}$.....V_1V_0> and 16K-byte pages. Assume that a TLB has 256 entries.

1) Show how the bits of a virtual address are used to access the TLB if the TLB is fully associative. Do the same if the TLB is 4-way set associative.

2) How many bytes of virtual memory are mapped by the fully associative TLB? By the 4-way set associative TLB?

3) Suppose you wanted to support 2 different page sizes: a 16Kbyte page size and a 1Mbyte page size, in the **fully-associative TLB**. Discuss options and trade-offs for both modifying the TLB or adding additional structures/mechanisms.

4) Suppose you wanted to support 2 different page sizes: a 16Kbyte page size and a 1Mbyte page size, in the **4-way set-associative TLB**. Discuss options and trade-offs for both modifying the TLB or adding additional structures/mechanisms.

2. Sharing Patterns & Implications

While shared memory programs can share information in arbitrary ways, there can be value in a taxonomy of sharing patterns, such as Weber and Gupta's classification of sharing into ***read-only, migratory, mostly-read, frequently read/written,*** and ***synchronization***.

1) Describe memory reference characteristics for each of Weber and Gupta's classification (or another classification of your choosing).

2) How might cache coherence protocols be adapted to target reference patterns from part (a)?

3) How might other memory systems aspects be adapted to target reference patterns from part (a)?

3. Branch prediction

1) Explain (with a diagram if you like) the working of the two-level branch prediction scheme as described by Yeh and Patt.

2) Imagine you are chief designer of the fetch unit for an 8-wide OOO processor i.e. a processor that must issue or rename 8 instructions every cycle. The fetch unit requires mechanisms beyond just a two-level branch predictor. Furthermore the fetch unit must fetch at least 8 instructions every cycle, which could pose challenges for the above predictor. Describe how you could employ the above predictor in the fetch unit and what additional structures and mechanisms might you decide to add to the fetch unit.

4. Reliability

1) What are transient errors or soft errors?

2) It is claimed that all microarchitectural structures are not equally susceptible to the effects of transient errors. Why? Discuss by comparing and contrasting the susceptibility of the branch predictor, load-store queue, register file, and write-through level-one data cache.

3) Discuss what techniques might be appropriate for mitigating the varied effects of transient errors in the branch predictor, load-store queue, register file, and write-through level-one data cache.

5. Dataflow vs. Out-of-order

Consider data-flow processing as used in the MIT Tagged Token Dataflow Architecture and conventional out-of-order processors like the MIPS R10K or other classes of processors like Pentium-Pro, Intel Nehalem etc.

1) What are the **similiarities** in the internal execution of the processor's instructions and the ordering of the instructions between the MIT Tagged Token Dataflow Architecture and OOO processors.

2) What are the **differences** in the internal execution of the processor's instructions and the ordering of the instructions between the MIT Tagged Token Dataflow Architecture and OOO processors.

6. Special vs. General Purpose Processors

Computer designers have for long proposed special-purpose architectures that are very effective at carrying out a specific function, but are less adept than general-purpose processors for running of variety of different applications. Today, with semiconductor technology providing plentiful transistor resources, many are proposing to integrate special-purpose accelerators alongside general-purpose processors. Others are arguing against special-purpose accelerators.

1) Argue why and under what conditions, if any, special-purpose accelerators are a good idea going forward.

2) Argue why and under what conditions, if any, special-purpose accelarators are not a good idea going forward.