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

Computer Architecture
Qualifying Examination

Fall 2020

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. Data Caches

The performance of a data cache design has, for many years, been studied using Hill's 3C model, where the 3 C's stand for *compulsory*, *capacity*, and *conflict* misses. Later, this model was extended to add a 4th C, *coherence* misses, in multiprocessor caches.

1. Briefly describe the above 4 different types of misses and why they occur.
2. Which one of the above 4 types of misses would you consider the least important to address in a cache design? Why?
3. Excluding the one type of miss of part (2), for the remaining 3 different types of misses, briefly describe 1-3 important techniques to reduce the type of misses.

2. Dealing with Branches

Handling conditional branching is seen as very important for high-performance microprocessors. However, different ISAs have taken significantly different approaches to providing support for handling conditional branching. For example, IA-64 and ARM support predication in the ISA, which in effect eliminates conditional branch instructions. Other ISAs have conditional branch instructions and rely on the microarchitecture to employ dynamic branch prediction for high performance.

You are the lead architect for a company that builds high performance microprocessors, and dealing with branches is becoming increasingly important. Assume that your company has a fixed ISA that already has support for both predication and conditional branches. Your co-worker Bob argues that your next processor should focus on predication, while your co-worker Carol argues that you should focus on branch prediction.

1. What arguments might Bob be making?
2. What arguments might Carol be making?
3. You are responsible for making the final decision on which path (or paths) to pursue. What will you do? Justify your answer.

3. GPU/Multi-Core

In recent years, as single-core performance stagnated, companies responded by designing multi-core CPUs. In addition to multi-core CPUs, general-purpose GPUs (GPGPUs) have also become increasingly popular with a large number of simpler "cores", referred to as "streaming multiprocessors" (also referred to as "compute units" by other GPU manufacturers).

1. The heart of the GPGPU microarchitecture is the streaming multiprocessor (SM). Describe 3 key features of the SM that help GPGPUs provide high performance.

2. What challenges (if any) do you see in continued performance improvements from GPGPUs with technology scaling and future transistor nodes? Justify your answer.

4. Cache Coherence

Cache coherence in multicore systems makes programming easy while introducing hardware complexity. Eliminating cache coherence has many times been proposed as a way to improve the power efficiency of processors with large core counts. Most systems even today provide robust support for cache coherence.

1. Explain the major sources of power inefficiency that are introduced by a snooping cache coherent system.
2. Explain how a directory-based coherence protocol alleviates some of these power overheads.
3. If low power is an important design objective, in addition to the other design objectives (e.g., performance, programmability), would you eliminate hardware cache coherence? Support your answer with a detailed reasoning.

5. Speculative Lock Elision and Transactional Memory

Intel and IBM have built processors in the past decade that implement *best-effort* hardware transactional memory. Best effort means that the transactions can always fail (in fact, a legal implementation of a begin transaction instruction is a noop). Even though transactions are not guaranteed to succeed, this type of hardware will enable programmers to exploit the concept of speculative lock elision.

1. Explain what speculative lock elision is and how it can increase concurrency in a parallel or multi-threaded shared memory program.
2. Explain why best-effort transactional memory hardware is easier to implement than other hardware transactional memory systems.
3. What limitations and pitfalls do best-effort transactional memory systems pose?

6. Out-Of-Order Processor Design

You are the chief architect at a company that has successfully designed a 4-issue out-of-order processor with an instruction window size of 160 instructions.

Your company leadership feels that it is important to have processors with the best single-thread performance without significant software assistance and thus has charged you

with designing an 8-issue out-of-order processor with a *very large instruction window size* (a window size of perhaps a 1000 instructions).

1. Enumerate the list of microarchitecture subsystems that are going to be critical for this large-window, 8-way design.
2. For each subsystem, identify the key challenges and some potential solutions.
3. Discuss some examples of how some of the design decisions in different subsystems influence each other.