**FALL 1994**
**COMPUTER SCIENCES DEPARTMENT**
**UNIVERSITY OF WISCONSIN—MADISON**
**PH. D. QUALIFYING EXAMINATION**

Computer Architecture
Qualifying Examination

Monday, September 19, 1994
3:30 – 7:30 PM
1240 Computer Sciences

**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 nontrivial.

**1. Fast Multipliers**

You are to design the logic for a high-speed multiplier to multiply two 8-bit numbers. (High-speed multipliers typically use carry-save adders.)

(a) Present the design of such a multiplier. In your design show the logic-level design of each building block, and how the building blocks are used to construct the final multiplier. (It is sufficient to specify the logic-level design using Boolean equations.)

(b) Estimate the total number of gates (which type, and how many inputs) in your multiplier.

(c) Estimate the delay through your multiplier, in terms of gate delays.

**2. Disk Technologies**

Spin-Quick Disk Industries is perfecting their design for a new 1.2 inch diameter hard disk drive for lap-top and palm-top computers. This new disk has one double-sided platter, with 2,000 tracks per inch and 20,000 bits per inch (per track). The drive motor hub occupies the center of the platter and has a diameter of 0.6 inches.

For this disk, answer the following questions. Your answers can be arithmetic expressions, or reduced to a single number.

(a) Assume the sector descriptor (sector number) requires 150 bits, the minimum interrecord gap is 200 bits, the sector data size is 4 kilobytes, and the error correcting code for the sector data is 48 bytes. How many sectors can fit on the innermost track? On the outermost track?

(b) What is the capacity of this disk if all tracks have the same number of sectors? ESTIMATE the capacity of the disk if instead it uses "constant bit density", which allows the number of sectors to vary from track to track.

(c) Assume that when seeking, it takes 2 ms to start the head moving, 0.02 ms per track, and 2 ms to stop and stabilize the head once it arrives at the destination track. What is the mean seek time assuming uniform random accesses?

(d) How fast must the disk spin to achieve a mean rotational latency of 2 ms (assume uniform random accesses)?

(e) Given your answers to parts (c) and (d), what is the total mean access time to transfer a 4 kilobyte sector? INCLUDE ONLY THE FACTORS DISCUSSED SO FAR.

**3. Bus Protocols**

A snooping bus for providing multiprocessor communication may be either *single-transaction* or *split-transaction*. A single-transaction bus permits only a single operation at a time, keeping the bus unavailable during the period when a read request is being serviced by the memory. A split-transaction bus permits multiple transactions to be outstanding by splitting a read request into two parts. A read request is followed by a release of the bus. Later, when the memory is prepared to return the result, it again arbitrates for the bus, acquiring it just long enough to send the requested data to the processor.

The split-transaction bus may be further refined by whether or not results are returned in the same order that they are requested. An *in-order* split-transaction bus always requires that read requests be completed in the same order in which they are initiated, while an *out-of-order* split-transaction bus places no restriction on the ordering of read requests. All split-transaction buses may have a limit on the number of concurrent oustanding reqeusts permitted.

(a) Compare latency and throughput characteristics for the three kinds of buses.

(b) Compared to a single-transaction bus, is it required that extra information be transmitted for an in-order split-transaction bus? For an out-of-order split-transaction bus?

(c) Does a cache-coherence algorithm need to be modified if it is to be used with a split-transaction bus? If so, explain how.

**4. Branch Instructions**

The definition of control instructions has been fiercely debated over many years. Some computers have simple compare_and_branch instructions. Others separate the compare and branch into two instructions, connecting them by the use of some state information (e.g., in a general register or condition codes). Recently some computers even have multiple sets of condition codes.

(a) Define the basic issues that must be considered in evaluating the relative merits of these variations in branch instructions.

(b) Give the arguments for separating compare and branch operations into two separate instructions vs. merging them into a single instruction.

(c) How are the branch methods related to the expected use of instruction pipelining in an implementation?

**5. Hardware support for multiprocessor synchronization**
Parallel machines often provide hardware support explicitly to facilitate interprocessor synchronization.

(a) Describe the functionality, intended usage, and implementation of the synchronization hardware provided by the following machines:

i) the Sequent Balance (the SLIC chip),

ii) the Thinking Machines CM-5 (the control network),

iii) the NYU Ultracomputer.

(b) Compare and contrast the utility of these mechanisms for different programming paradigms, for example, shared memory and data parallel.

**6. Cache Memories**
Caches are typically indexed with the least significant address bits of the block number and use the higher order bits for tags. Computers with paged virtual memory translate virtual addresses into physical addresses. Caches in such computers can use virtual addresses for both tags and indexes (VT/VI) or physical addresses for both (PT/PI).

(a) Compare and contrast VT/VI and PT/PI caches. Be sure to consider in your discussion the effect of cache size, context switches, whether the architecture includes address space identifiers, and whether the cache must support multiprocessor coherence.

(b) Two hybrid caches are also possible: physically-tagged/virtually-indexed (PT/VI) and virtually-tagged/physically-indexed (VT/PI). When might they be used? Compare and contrast them with VT/VI and PT/PI caches and each other.