Fall 1998 Systems Qualifying Exam
Computer Science Dept., Indiana University

Computer Architecture

We have a proposal for three different machines, M0, M8 and M16, that differ in their register count. All three machines have three-operand instructions, and any operand can be a memory reference or a register (except for M0, which doesn't have any registers).

For example, here is an ADD instruction for M8:

ADD R1, R6, COUNTER

COUNTER is a memory reference.

The cost of a memory operand on each machine is six cycles, and the cost of a register operand is one cycle. Each of the three operands has equal probility of being in a register.

The differences between these machines is described in the following table. The execution cycles are IN ADDITION to the cost of operand access. The probability of an operand being in a register as opposed to memory applies to each operand individually.

Machine	Register Count	Execution Cycles	Probability Operand in Register
M0	0	4	0.0
M8	8	5	0.5
M16	16	6	0.8

1. (20 points) What is the cycle count for an average instruction on each machine? Justify your computed answer and show your work.

2. (40 points) Each machine was initially designed without pipelining. The second version of each machine was fully pipelined but NOT superscalar. Diagram the schematic of each machine's pipeline. What is the performance improvement for the pipelined version of each machine compared to the unpipelined version? Justify your computed answer and show your work.

3. (20 points) Make a table comparing the data hazards that affect the pipelined version of each machine.

4. (20 points) Using the table you constructed in question 3, give at least two hardware and two software methods to deal with the data hazards. If one or more methods do NOT apply to a particular machine, explain why not.

Operating Systems

Problem 1. Define multi-tasking (also called multi-programming) and multi-threading. What are the benefits of each? For multi-tasking, give a order-of-magnitude quantitive estimate of the increase in CPU utilization compared to a system that does not support multi-tasking (or multi-threading); to simplify this calculation, assume that a typical program performs a disk I/O operation every 10^6 machine instructions (on average), and that other I/O operations are much rarer. Clearly state your estimates for typical values of other relevant system parameters.

Problem 2. Give pseudo-code for an implementation of locks. The only synchronization primitive you may use is semaphores. Your implementation should work correctly even on a multi-processor. If a thread calls Release(ℓ) for a lock ℓ that the thread does not hold, or if a thread calls Acquire(ℓ) for a lock ℓ that the thread already holds, your implementation should return -1 (or raise an exception) to indicate that an error occurred.

Problem 3. (a) What replacement policy is typically used for block caches in file systems? Briefly describe (in English or pseudo-code) an efficient implementation of this replacement policy. (b) Is this policy optimal? Justify your answer. (c) Is this policy also used for page replacement in demand-paged virtual memory systems? Why or why not?

Problem 4. Distributed file systems typically cache recently accessed blocks of remote files. Suppose a process writes data to a fragment of a block that is not currently in the local cache; for example, in a file system with a block size of 8KB, the process might write 1KB of data at offset 2048 in some block. Describe at least three ways in which the cache might handle such writes. Discuss the advantages and disadvantages of each.