CpE 313
Microprocessor System Design
Solution to Quiz 3

Instructions

Read each individual problem appearing on this quiz carefully and do only what is specifically stated. Each question is worth 10 points.

This quiz is designed to be completed by a well-prepared student in approximately **30 minutes**; most students are expected to finish it within the allotted time. On your initial pass through this quiz, skip any problems that appear to be overly difficult.

IMPORTANT: Write your initials at the TOP of EACH page. Also, be sure to read and sign the Academic Honesty Statement that follows:
"In signing this statement, I hereby certify that the work on this quiz is my own and that I have not copied the work of any other student while completing this quiz. I understand that, if I fail to honor this agreement, I will receive a score of ZERO for this quiz and will be subject to possible disciplinary action."

Printed Name: .... **Signature:** ....

Date: ....

DO NOT BEGIN UNTIL INSTRUCTED TO DO SO.

Virtual Memory

Problem 1.

(a) While caches usually have low associativity, memory is usually always fully associative. Why is this the case?

Response Example 1: *Because more than 95% of the memory references hit in the cache, it is very important to have a very low hit time in caches. Because a lower associativity reduces hit time, caches usually have lower associativities.*
The access time ratio is about 25 for L1 cache to memory, but is about 10^4 *for memory to disk. That is, a miss in memory is much more costly than a miss in L1 cache. Because lower associativity may increase conflict misses, the memory is usually fully associative to reduce misses or page faults.*

(b) What is a page table? How is it used to implement virtual memory?

Response Example 1: *The page table for a process is a data structure that stores the physical location, memory or disk, of all virtual pages of that process.*
It implements virtual memory by translating a virtual page number to its corresponding physical page number if the page is in the memory, OR generating a page fault if the page is on the disk.

Translation Lookaside Buffer

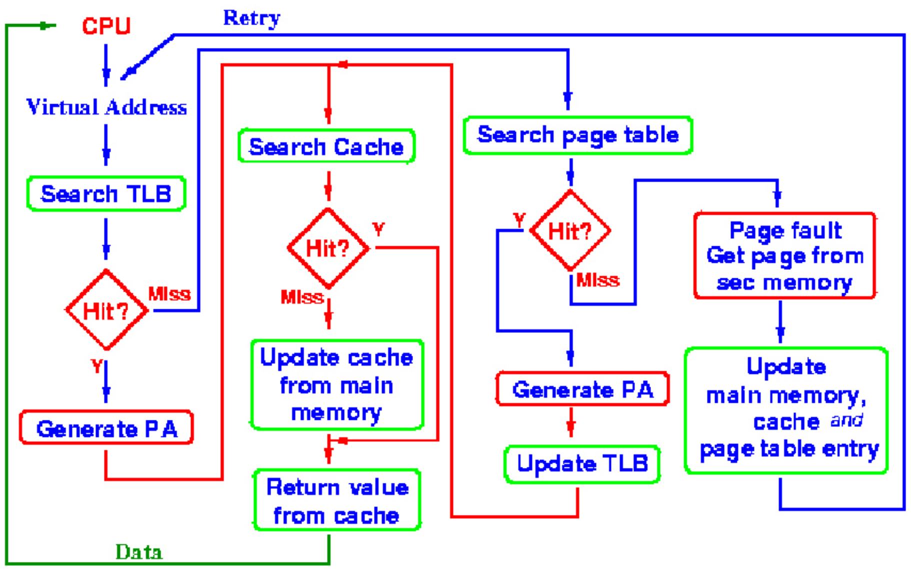
Problem 2.

(a) Why is a TLB flushed after a context switch? Given that a TLB is nothing but a few cached page table entries, why is a page table not "flushed" after a context switch?

Response Example 1: *Assumption: The TLB in question does not store PIDs. The TLB must be flushed after a context switch because each process has different VA→PA mappings making the TLB entries useless after a process switch. The page table isn't "flushed" because there is an entirely separate page table for each process, therefore it switches automatically.* ***(Adapted from Craig's response.)***

Response Example 2: *TLBs that do not include PID info must be flushed after a context switch because every program runs in a memory that is virtually addressed starting at 0 (i.e., virtual addresses are duplicated). Page tables do not need flushing because each process has its own table,and that table is automatically changes during a context switch* ***(Adapted from Jordan's response.)***

For the next two parts of this question (on the next page), consult the figure below. Assume that you have a P/P cache.



More Translation Lookaside Buffer

Problem 2 (cont'd).
(b) Following the generation of an address by the CPU, is it possible to have a hit in the TLB, a miss in the cache, and a hit in the page table? Why or why not?

Response Example 1: *YES. This will happen when the page is stored in memory (RAM), hence the page table contains the VA→PA mapping (a hit). Also TLB has a copy of that translation from page table if the page was recently accessed (a hit). But the cache block corresponding to this VA is not present in the cache because it was never accessed before (a miss), or was accessed but got kicked out (a miss).* ***(Adapted from Maciej's response.)***

Response Example 2: *Yes, but the page table only needs to be searched for the translation if there is a miss in the TLB. However, since the TLB is updated when there is a hit in the page table, it will only hit in the TLB if it hits in the page table (or, equivalently, it will not hit in the TLB if it will not hit in the page table).* ***(Adapted from Dan's response.)***

Response Example 3: *Yes. A previous access may have loaded the entry into the TLB, and later use may clear its cache line from the cache. As long as the page has been bumped from the memory, its TLB entry remains valid. This case allows a TLB hit and cache miss, with an implied page table hit (although this is not checked if TLB hits).* ***(Adapted from Jordan's response.)***

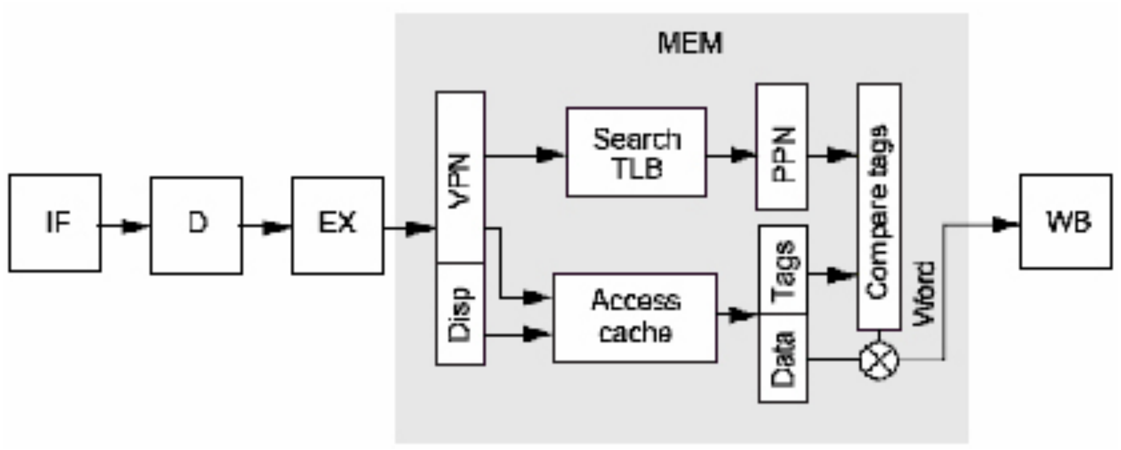
(c) Following the generation of an address by the CPU, is it possible to have a hit in the TLB, a miss in the cache, and a miss in the page table? Why or why not?

Response Example 1: *No. Because TLB contains only entries copied from page table. Whenever a page table entry is updated, the memory controller updates the TLB to ensure that if an entry is not in the page table, it cannot exist in the TLB.*

Virtually Indexed Caches

Problem 3.

(a) For the organization shown below, what is the maximum 2-way associative cache size if the page size is 8KB? What really limits the cache size for this organization?



Response Example 1: *If the index of a cache is restricted to bits within page offset, only then the cache size is restricted to be one page size per way of the associativity. However, because the cache in question does not have this restriction, its size is not limited by the page size. What really limits the size is the time it will take to search a large cache and the cost of building a large cache on the processor chip.*