CMU 18-447 Introduction to Computer Architecture, Spring 2014
HW 5: Virtual Memory, Caching and Main Memory

Instructor: Prof. Onur Mutlu
TAs: Rachata Ausavarungnirun, Varun Kohli, Xiao Bo Zhao, Paraj Tyle

Assigned: Wed., 3/11, 2014
Due: **Fri., 3/28, 2014 (Midnight)**
Handin: `/afs/ece/class/ece447/handin/hw5`
Please submit as ONE PDF: [andrewID].pdf

1 Virtual Memory [20 points]

An ISA supports an 8-bit, byte-addressable virtual address space. The corresponding physical memory has only 128 bytes. Each page contains 16 bytes. A simple, one-level translation scheme is used and the page table resides in physical memory. The initial contents of the frames of physical memory are shown below.

Frame Number	Frame Contents
0	Empty
1	Page 13
2	Page 5
3	Page 2
4	Empty
5	Page 0
6	Empty
7	Page Table

A three-entry translation lookaside buffer that uses Least Recently-Used (LRU) replacement is added to this system. Initially, this TLB contains the entries for pages 0, 2, and 13. For the following sequence of references, put a circle around those that generate a TLB hit and put a rectangle around those that generate a page fault. What is the hit rate of the TLB for this sequence of references? (Note: LRU policy is used to select pages for replacement in physical memory.)

References (to pages): 0, 13, 5, 2, 14, 14, 13, 6, 6, 13, 15, 14, 15, 13, 4, 3.

Solution:
References (to pages): (0), (13), 5, 2, [14], (14), 13, [6], (6), (13), [15], 14, (15), (13), [4], [3].

TLB Hit Rate = 7/16

(a) At the end of this sequence, what three entries are contained in the TLB?
Solution:
4, 13, 3

(b) What are the contents of the 8 physical frames?
Solution:
Pages 14, 13, 3, 2, 6, 4, 15, Page table

2 Page Table Bits [5 points]

(a) What is the purpose of the "reference" or "accessed" bit in a page table entry?

Solution:

To aid page replacement.

(b) Describe what you would do if you did not have a reference bit in the PTE. Justify your reasoning and/or design choice.

Solution:

Pick a random page to replace when a page fault occurs.

(c) What is the purpose of the dirty or modified bit in a page table entry?

Solution:

To enable writeback of only dirty pages (rather than all pages) to disk.

(d) Describe what you would do if you did not have a modified bit in the PTE. Justify your reasoning and/or design choice.

Solution:

Write back all pages to disk.

Alternative answer: the OS could map all pages read-only by default. On a page-fault due to a write, if the program has permission to change the page, the operating system remaps the page as read-write and also knows that the page has become dirty.

3 Caching [15 points]

Below, we have given you four different sequences of addresses generated by a program running on a processor with a data cache. Cache hit ratio for each sequence is also shown below. Assuming that the cache is initially empty at the beginning of each sequence, find out the following parameters of the processor's data cache:

- Associativity (1, 2 or 4 ways)
- Block size (1, 2, 4, 8, 16, or 32 bytes)
- Total cache size (256 B, or 512 B)
- Replacement policy (LRU or FIFO)

Assumptions: all memory accesses are one byte accesses. All addresses are byte addresses.

Sequence No.	Address Sequence	Hit Ratio
1	0, 2, 4, 8, 16, 32	0.33
2	0, 512, 1024, 1536, 2048, 1536, 1024, 512, 0	0.33
3	0, 64, 128, 256, 512, 256, 128, 64, 0	0.33
4	0, 512, 1024, 0, 1536, 0, 2048, 512	0.25

Solution:
Cache block size - 8 bytes
For sequence 1, only 2 out of the 6 accesses (specifically those to addresses 2 and 4) can hit in the cache, as the hit ratio is 0.33. With any other cache block size but 8 bytes, the hit ratio is either smaller or larger than 0.33.
Therefore, the cache block size is 8 bytes.

Associativity - 4

For sequence 2, blocks 0, 512, 1024 and 1536 are the only ones that are reused and could potentially result in cache hits when they are accessed the second time. Three of these four blocks should hit in the cache when accessed for the second time to give a hit rate of 0.33 (3/9).

Given that the block size is 8 and for either cache size (256B or 512B), all of these blocks map to set 0. Hence, an associativity of 1 or 2 would cause at most one or two of these four blocks to be present in the cache when they are accessed for the second time, resulting in a maximum possible hit rate of less than 3/9. However, the hit rate for this sequence is 3/9. Therefore, an associativity of 4 is the only one that could potentially give a hit rate of 0.33 (3/9).

Total cache size - 256 B
For sequence 3, a total cache size of 512 B will give a hit rate of 4/9 with a 4-way associative cache and 8 byte blocks regardless of the replacement policy, which is higher than 0.33. Therefore, the total cache size is 256 bytes.

Replacement policy - LRU
For the aforementioned cache parameters, all cache lines in sequence 4 map to set 0. If a FIFO replacement policy were used, the hit ratio would be 3/8, whereas if an LRU replacement policy were used, the hit ratio would be 1/4. Therefore, the replacement policy is LRU.

4 Magic-RAM [25 points]

Assume you developed the next greatest memory technology, MagicRAM. A MagicRAM cell is non-volatile. The access latency of a MagicRAM cell is 2 times that of an SRAM cell but the same as that of a DRAM cell.

The read/write energy of MagicRAM is similar to the read/write energy of DRAM. The cost of MagicRAM is similar to that of DRAM. MagicRAM has higher density than DRAM. MagicRAM has one shortcoming, however: a MagicRAM cell stops functioning after 2000 writes are performed to the cell.

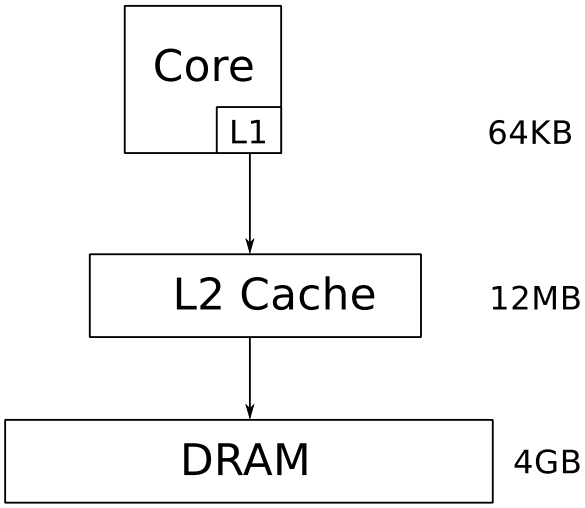
(a) Is there an advantage of MagicRAM over DRAM other than its density? (Please do not repeat what is stated in the above paragraph.) Circle one: **YES** **NO**

Explain.

(b) Is there an advantage of MagicRAM over SRAM? Circle one: **YES** **NO**

Explain.

(c) Assume you have a system that has a 64KB L1 cache made of SRAM, a 12MB L2 cache made of SRAM, and 4GB main memory made of DRAM.



Assume you have complete design freedom and add structures to overcome the shortcoming of MagicRAM. You will be able to propose a way to reduce/overcome the shortcoming of MagicRAM (note that you can design the hierarchy in any way you like, but cannot change MagicRAM itself).

Does it makes sense to add MagicRAM anywhere in this memory hierarchy given that you can potentially reduce its shortcoming? Circle one: **YES** **NO**

If so, where would you place MagicRAM? **Depict in the figure above clearly** and describe why you made this choice.

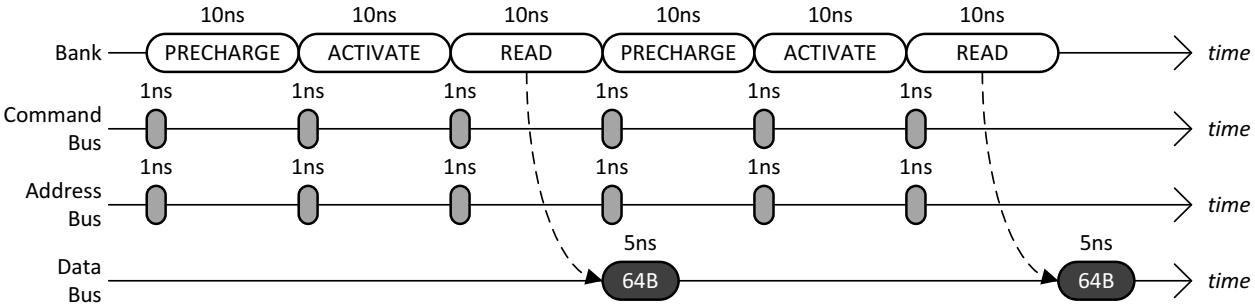
If not, why not? Explain below clearly and methodically.

(d) Propose a way to reduce/overcome the shortcoming of MagicRAM by modifying the given memory hierarchy. Be clear in your explanations and illustrate with drawings to aid understanding.

Explanation:

5 Memory Scheduling (Optional)

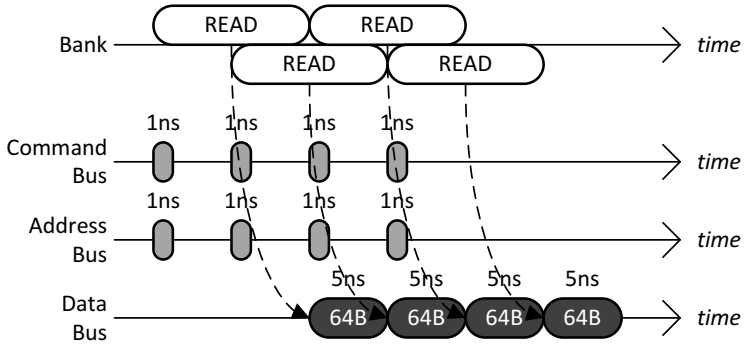
Row-Buffer Conflicts. The following timing diagram shows the operation of a single DRAM channel and a single DRAM bank for two back-to-back reads that conflict in the row-buffer. Immediately after the bank has been busy for 10ns with a `READ`, data starts to be transferred over the data bus for 5ns.



(a) Given a long sequence of back-to-back reads that always conflict in the row-buffer, what is the data throughput of the main memory system? Please state your answer in **gigabytes/second**.

(b) To increase the data throughput, the main memory designer is considering adding more DRAM banks to the single DRAM channel. Given a long sequence of back-to-back reads to all banks that always conflict in the row-buffers, what is the minimum number of banks that is required to achieve the maximum data throughput of the main memory system?

Row-Buffer Hits. The following timing diagram shows the operation of the single DRAM channel and the single DRAM bank for four back-to-back reads that hit in the row-buffer. It is important to note that row-buffer hits to the same DRAM bank are pipelined: while each `READ` keeps the DRAM bank busy for 10ns, up to at most **half** of this latency (5ns) can be overlapped with another read that hits in the row-buffer. (Note that this is different from Lab 6 where we unrealistically assumed that row-buffer hits are non-pipelined.)



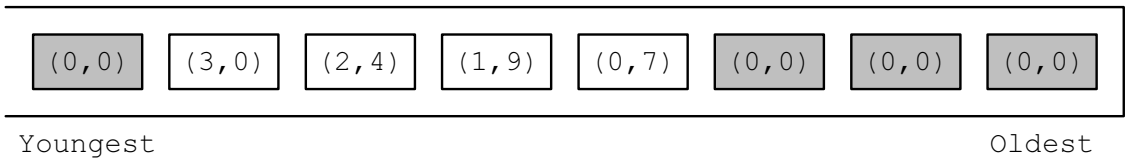
(c) Given a long sequence of back-to-back reads that always hits in the row-buffer, what is the data throughput of the main memory system? Please state your answer in **gigabytes/second**.

(d) When the maximum data throughput is achieved for a main memory system that has a single DRAM channel and a single DRAM bank, what is the bottleneck that prevents the data throughput from becoming even larger? **Circle** all that apply.

BANK COMMAND BUS ADDRESS BUS DATA BUS

Memory Scheduling Policies. The diagram below shows the memory controller's *request queue* at time 0. The shaded rectangles are read requests generated by thread $T0$, whereas the unshaded rectangles are read requests generated by thread $T1$. Within each rectangle, there is a pair of numbers that denotes the request's (*BankAddress*, *RowAddress*). Assume that the memory system has a **single** DRAM channel and **four** DRAM banks. Further assume the following.

- All the row-buffers are **closed** at time 0.
- Both threads start to stall at time 0 because of memory.
- A thread continues to stall until it receives the data for all of its requests.
- Neither thread generates more requests.



For extra credits (15 points), please make sure that you model contention in the banks as well as in all of the buses (address/command/data).

(e) For the `FCFS` scheduling policy, calculate the memory stall time of $T0$ and $T1$.

(f) For the `FR-FCFS` scheduling policy, calculate the memory stall time of $T0$ and $T1$.

(g) For the `PAR-BS` scheduling policy, calculate the memory stall time of $T0$ and $T1$. Assume that all eight requests are included in the same *batch*.

6 Cache Enigma [35 points]

A processor has a 4-way set-associative L1 cache that can house 4 blocks in total. The access latency to this cache is 1 cycle. The replacement policy is true LRU. The processor is known to not employ any prefetching mechanism.

The processor also has a 16-way set-associative L2 cache that can house 128 blocks in total. The access latency to this cache is 20 cycles.

A programmer writes a test program that in a loop repeatedly accesses only the following data cache blocks (assume billions of iterations are run):

A, B, C, D, E, F

where A, ..., F are different cache block addresses.

In the steady state (i.e., after the loop has executed for a few iterations), the programmer finds out that the average memory access time is 1 cycle.

Then, the programmer writes another program that in a loop repeatedly accesses only the following data cache blocks:

A, B, C, D, E, F, G, H

In the steady state (i.e., after the loop has executed for a few iterations), the programmer finds out that the average memory access time is 20 cycles.

(a) What can you say about this processor? (I.e., what is going on?)

Please describe everything you can say, concretely, but be concise.

The L1 cache can hold only 4 blocks. Therefore, a victim cache with an access latency of 1 cycle is present. This victim cache is accessed in parallel with the access to L1 cache. Cache blocks evicted from the L1 cache are inserted into the victim cache. Based on the access latencies from above, the victim cache can hold either 2 or 3 cache blocks.

(b) Based on the above information, what do you expect the average memory access time of yet another program that in a loop repeatedly accesses only the following data cache blocks?

A, B, C, D, E

Explain:

1 cycle. The 5 cache blocks from above can all fit in the L1 cache and victim cache, and thus all cache block accesses are hits (in the steady state).

(c) Again, based on the above information, what do you expect the average memory access time of yet another program that in a loop repeatedly accesses only the following data cache blocks?

A, B, C, D, E, F, G

Explain:

Either 1 or 20 cycles.
If the victim cache holds 2 blocks, the 7 cache blocks from above cannot all fit in the two caches, resulting in all accesses being misses.
If the victim cache holds 3 blocks, the 7 cache blocks from above can all fit in the two caches, resulting in all accesses being hits.

(d) Finally, again, based on the above information, what do you expect the average memory access time of yet another program that in a loop repeatedly accesses only the following data cache blocks?

A, B, C, D, E, F, G, H, I

Explain:

20 cycles. It is impossible for all 9 cache blocks to fit in the two caches, so all accesses are misses in the steady state.

7 Cache and Virtual Memory [40 points]

A four-way set-associative writeback cache has a $2^{11} \cdot 89$-bit tag store. The cache uses a custom replacement policy that requires 9 bits per set. The cache block size is 64 bytes. The cache is virtually-indexed and physically-tagged. Data from a given physical address can be present in up to eight different sets in the cache. The system uses hierarchical page tables with two levels. Each level of the page table contains 1024 entries. A page table may be larger or smaller than one page. The TLB contains 64 entries.

tag store: 89 bits -¿ (9 replacement, 4 dirty, 4 valid) + 4 * (18 bit phys tag)

2^{11} -¿ 2K sets -¿ 8K blocks -¿ 512KB cache

phys addr has 8 synonyms –¿ 3 bits of set number are from VPN/PFN –¿ 16K pages

virt address: — — set (11 bits) — block offset (6 bits) — phys address: — PFN (18 bits) — page offset (14 bits) —

1024^2 -¿ 2^{20} pages. Thus 16 GB virtual address space

tag has 18 Phys Addr bits, plus 14 page bits -¿ 2^{32} physaddr -¿ 4 GB physical address space

(a) How many bits of the virtual address are used to choose a set in the cache?

11

(b) What is the size of the cache data store?

512KB

(c) How many bits in the Physical Frame Number must overlap with the set index bits in the virtual address?

3

(d) On the following blank figure representing a virtual address, draw in bitfields and label bit positions for "cache block offset" and "set number." Be complete, showing the beginning and ending bits of each field.

Virtual Address:

(e) On the following blank figure representing a physical address, draw in bitfields and label bit positions for "physical frame number" and "page offset." Be complete, showing the beginning and ending bits of each field.

Physical Address:

(f) What is the page size?

16 KB

(g) What is the size of the virtual address space?

16 GB

(h) What is the size of the physical address space?

4 GB