

CMU 18-447 INTRODUCTION TO COMPUTER ARCHITECTURE, SPRING 2013

HW 6: CACHING AND MEMORY

Instructor: Prof. Onur Mutlu

TAs: Justin Meza, Yoongu Kim, Jason Lin

Assigned: Wed., 4/3, 2013

Due: **Mon., 4/22, 2013 (Midnight)**

Handin: /afs/ece/class/ece447/handin/hw6

**Please submit as ONE PDF: [andrewID].pdf**

## 1 Virtual Memory and Caching [10 points]

A 2-way set associative write back cache with perfect LRU replacement requires  $15 \times 2^9$  bits of storage to implement its tag store (including bits for valid, dirty and LRU). The cache is virtually indexed, physically tagged. The virtual address space is 1 MB, page size is 2 KB, cache block size is 8 bytes.

- What is the size of the data store of the cache in bytes?
- How many bits of the virtual index come from the virtual page number?
- What is the physical address space of this memory system?

## 2 Memory Interleaving [15 points]

A machine has a main memory of 4 KB, organized as 1 channel, 1 rank and  $N$  banks (where  $N > 1$ ). The system does not have virtual memory.

- Data is interleaved using a cache block interleaving policy, as described in lecture, where consecutive cache blocks are placed on consecutive banks.
  - The size of a cache block is 32 bytes. Size of a row is 128 bytes.
  - An open row policy is used, i.e., a row is retained in the row-buffer after an access, until an access to another row is made.
  - A row-buffer hit is an access to a row that is present in the row-buffer. A row-buffer miss is an access to a row that is not present in the row-buffer.
- For a program executing on the machine, accesses to the following bytes miss in the on-chip caches and go to memory.

0, 32, 320, 480, 4, 36, 324, 484, 8, 40, 328, 488, 12, 44, 332, 492

The row-buffer hit rate is 0%, i.e., all accesses miss in the row-buffer.

What is the minimum value of  $N$  - the number of banks?

- If the row-buffer hit rate for the same sequence were 75%, what would be minimum value of  $N$  - the number of banks?
- i) Could the row-buffer hit rate for the sequence be 100%? Why or why not? Explain.  
ii) If yes, what is the minimum number of banks required to achieve a row-buffer hit rate of 100%?

### 3 More Memory Interleaving [20 points]

A DRAM main memory system has  $N$  banks in one rank on one channel. A row is 256 bytes and a cache block is 64 bytes. Data is row-interleaved across banks using the following physical address bit scheme:

Row	Bank	Column	Byte in bus
-----	------	--------	-------------

An open row policy is used, i.e., a row is retained in the row-buffer after an access until an access to another row is made. Initially, row 1024 is open in all banks in all channels.

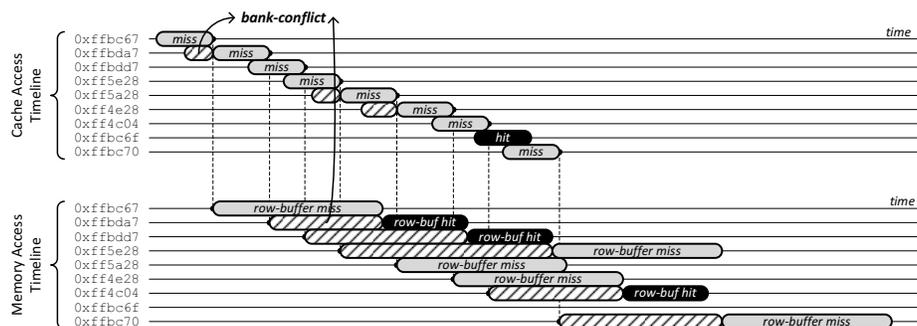
(a) Determine the total number of banks (across all channels) in the system given that the row-buffer hit rate on the following sequence of *cache block numbers* is 33.3% ( $\frac{1}{3}$ ):

0, 4, 8, 16, 32, 64, 128, 256, 128, 64, 32, 16, 8, 4, 0.

(b) For what total number of banks in the system will the row-buffer hit rate of this sequence be  $\frac{7}{15}$ ?

### 4 Banks [30 points]

A processor's memory hierarchy consists of a small SRAM L1-cache and a large DRAM main memory, both of which are banked. The processor has a 24-bit physical address space and does not support virtual memory (i.e., all addresses are physical addresses). An application has just started running on this processor. The following figure shows the timeline of memory references made by that application and how they are served in the L1-cache or main memory.



For example, the first memory reference made by the application is to byte-address `0xffbc67` (assume that all references are byte-sized reads to byte-addresses). However, the memory reference misses in the L1-cache (assume that the L1-cache is initially empty). Immediately afterwards, the application accesses main memory, where it experiences a row-buffer miss (initially, assume that all banks in main memory each have a row opened that will never be accessed by the application). Eventually, the cache block (and only that cache block) that contains the byte-address `0xffbc67` is fetched from memory into the cache.

Subsequent memory references may experience *bank-conflicts* in the L1-cache and/or main memory, if there is a previous reference still being served at that particular bank. Bank-conflicts are denoted as hatched shapes in the timeline.

The following table shows the address of the memory references made by the application in both hexadecimal and binary representations.

Hexadecimal	Binary
ffbc67	1111 1111 1011 1100 0110 0111
ffbda7	1111 1111 1011 1101 1010 0111
ffbdd7	1111 1111 1011 1101 1101 0111
ff5e28	1111 1111 0101 1110 0010 1000
ff5a28	1111 1111 0101 1010 0010 1000
ff4e28	1111 1111 0100 1110 0010 1000
ff4c04	1111 1111 0100 1100 0000 0100
ffbc6f	1111 1111 1011 1100 0110 1111
ffbc70	1111 1111 1011 1100 0111 0000

From the above timelines and the table, your job is to answer questions about the processor's cache and main memory organization. Here are some assumptions to help you along the way.

- Assumptions about the L1-cache
  - Block size: ? (Power of two, greater than two)
  - Associativity: ? (Power of two, greater than two)
  - Total data-store size: ? (Power of two, greater than two)
  - Number of banks: ? (Power of two, greater than two)
  - Initially empty
- Assumptions about main memory
  - Number of channels: 1
  - Number of ranks per channel: 1
  - Number of banks per rank: ? (Power of two, greater than two)
  - Number of rows per bank: ? (Power of two, greater than two)
  - Number of cache-blocks per row: ? (Power of two, greater than two)
  - Contains the entire working set of the application
  - Initially, all banks have their 0<sup>th</sup> row open, which is never accessed by the application

#### 4.1 First, let's cover the basics

- (a) Caches and main memory are sub-divided into multiple banks in order to allow parallel access. What is an alternative way of allowing parallel access?
- (b) A cache that allows multiple cache misses to be outstanding to main memory at the same time is called what? (Two words or less. Hint: It's an adjective.)
- (c) While cache misses are outstanding to main memory, what is the structure that keeps bookkeeping information about the outstanding cache misses? This structure often augments the cache.
- (d) Which is larger, an SRAM cell or a DRAM cell?
- (e) What is the number of transistors and/or capacitors needed to implement each cell, including access transistor(s)?

#### 4.2 Cache and memory organization

NOTE: For the following questions, assume that all offsets and indexes come from contiguous address bits.

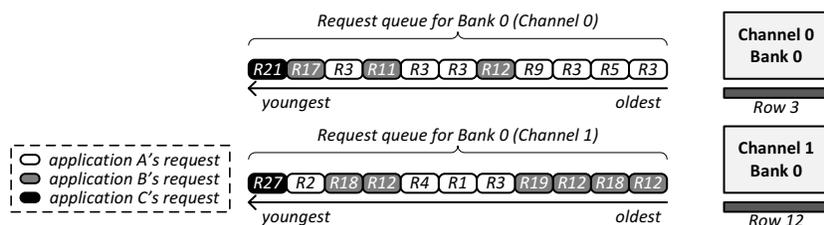
- What is the L1-cache's block size in bytes? Which bit positions in the 24-bit physical address correspond to the cache block offset? (The least-significant bit in the physical address has a bit position of 0.)
- How many banks are there in the L1-cache? Which bit positions in the 24-bit physical address correspond to the L1-cache bank index? (The least-significant bit in the physical address has a bit position of 0.)
- How many banks are there in main memory? Which bit positions in the 24-bit physical address correspond to the main memory bank index? (The least-significant bit in the physical address has a bit position of 0.)
- What kind of interleaving is used to map physical addresses to main memory?
- To fully support a 24-bit physical address space, how many rows must each main memory bank have? Which bit positions in the 24-bit physical address correspond to the main memory row index? (The least-significant bit in the physical address has a bit position of 0.)
- Each cache block within a row is called a *column*. How many columns are there in a single row? Which bit positions in the 24-bit physical address correspond to the main memory column index? (The least-significant bit in the physical address has a bit position of 0.)

## 5 Memory Scheduling [35 points]

To serve a memory request, the memory controller issues one or multiple DRAM commands to access data from a bank. There are four different DRAM commands.

- ACTIVATE:** Loads the row (that needs to be accessed) into the bank's row-buffer. This is called *opening* a row. (**Latency: 15ns**)
- PRECHARGE:** Restores the contents of the bank's row-buffer back into the row. This is called *closing* a row. (**Latency: 15ns**)
- READ/WRITE:** Accesses data from the row-buffer. (**Latency: 15ns**)

The following figure shows the snapshot of the memory request buffers (in the memory controller) at  $t_0$ . Each request is color-coded to denote the application to which it belongs (assume that all applications are running on separate cores). Additionally, each request is annotated with the address (or index) of the row that the request needs to access (e.g.,  $R_3$  means that the request is to the 3<sup>rd</sup> row). Furthermore, assume that all requests are read requests.



A memory request is considered to be *served* when the READ command is complete (i.e., 15ns after the request's READ command has been issued). In addition, each application (A, B, or C) is considered to be **stalled** until *all* of its memory requests (across all the request buffers) have been served.

Assume that, initially (at  $t_0$ ) each bank has the 3<sup>rd</sup> and the 12<sup>th</sup> row loaded in the row-buffer, respectively. Furthermore, no additional requests from any of the applications arrive at the memory controller.

## 5.1 Application-Unaware Scheduling Policies

- (a) Using the **FCFS** scheduling policy, what is the **stall time** of each application?
- (b) Using the **FR-FCFS** scheduling policy, what is the stall time of each application?
- (c) What property of memory references does the *FR-FCFS* scheduling policy exploit? (Three words or less.)
- (d) Briefly describe the scheduling policy that would **maximize** the *request throughput* at any given bank, where request throughput is defined as the number of requests served per unit amount of time. (Less than 10 words.)

## 5.2 Application-Aware Scheduling Policies

Of the three applications, application C is the least memory-intensive (i.e., has the lowest number of outstanding requests). However, it experiences the largest stall time since its requests are served only after the numerous requests from other applications are first served. To ensure the shortest stall time for application C, one can assign its requests with the highest priority, while assigning the same low priority to the other two applications (A and B).

- (a) **Scheduling Policy X:** When application C is assigned a high priority and the other two applications are assigned the same low priority, what is the stall time of each application? (Among requests with the same priority, assume that *FR-FCFS* is used to break ties.)

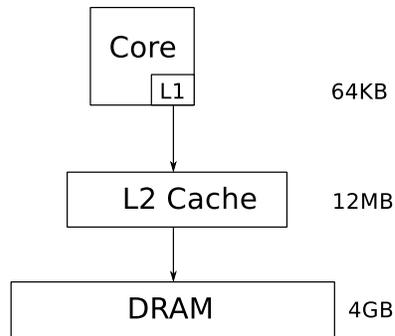
Can you design an even better scheduling policy? While application C now experiences low stall time, you notice that the other two applications (A and B) are still delaying each other.

- (b) Assign priorities to the other two applications such that you minimize the average stall time across all applications. Specifically, list all **three** applications in the order of highest to lowest priority. (Among requests with the same priority, assume that *FR-FCFS* is used to break ties.)
- (c) **Scheduling Policy Y:** Using your new scheduling policy, what is the stall time of each application? (Among requests with the same priority, assume that *FR-FCFS* is used to break ties.)
- (d) Order the four scheduling policies (FCFS, FR-FCFS, X, Y) in the order of lowest to highest average stall time.

## 6 Memory Hierarchy [20 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.) Explain.
- (b) Is there an advantage of MagicRAM over SRAM? 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).

(i) Does it makes sense to add MagicRAM anywhere in this memory hierarchy given that you can potentially reduce its shortcoming?

(ii) If so, where would you place MagicRAM? Describe it in terms of the figure above 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.