Instructor: Prof. Onur Mutlu
TAs: Mohammed Alser, Rahul Bera, Geraldo Francisco De Oliveira Junior, Can Firtina,
Juan Gomez Luna, Jawad Haj-Yahya, Hasan Hassan, Konstantinos Kanellopoulos, Jeremie Kim,
Nika Mansouri Ghiasi, Lois Orosa Nogueira, Jisung Park, Minesh Hamenbhai Patel, Abdullah Giray Yaglikci

Given: Thursday, Oct 31, 2019
Due: **Wednesday, Nov 14, 2019**

- **Handin - Critical Paper Reviews (1).** You need to submit your reviews to `https://safari.ethz.ch/review/architecture19/`. Please, check your inbox, you should have received an email with the password you should use to login. If you didn't receive any email, contact comparch@lists.ethz.ch. In the first page after login, you should click in "Architecture - Fall 2019 Home", and then go to "any submitted paper" to see the list of papers.
- **Handin - Questions (2-4).** You should upload your answers to the Moodle Platform (`https://moodle-app2.let.ethz.ch/mod/assign/view.php?id=382814`) as a single PDF file.

---

1. Critical Paper Reviews [300 points]

Please read the guidelines for reviewing papers and check the sample reviews. You may access them by *simply clicking on the QR codes below or scanning them*. We will give out extra credit that is worth 0.5% of your total grade for each good review.

Guidelines

Sample reviews

Write an approximately one-page critical review for each of the following papers. A review with bullet point style is more appreciated. Try not to use very long sentences and paragraphs. Keep your writing and sentences simple. Make your points bullet by bullet, as much as possible.

- Lee et al., "Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture," in Proceedings of the 19th International Symposium on High-Performance Computer Architecture (HPCA), 2013. `https://people.inf.ethz.ch/omutlu/pub/tldram_hpca13.pdf`
- Kim et al., "D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput," in Proceedings of the 25th International Symposium on High-Performance Computer Architecture (HPCA), 2019. `https://people.inf.ethz.ch/omutlu/pub/drange-dram-latency-based-true-random-number-generator_hpca19.pdf`
- Ipek et al., "Self Optimizing Memory Controllers: A Reinforcement Learning Approach," in Proceedings of the 35th International Symposium on Computer Architecture (ISCA), 2008. `https://people.inf.ethz.ch/omutlu/pub/rlmc_isca08.pdf`

2. RowClone [150 points]

Recall that the RowClone[1] idea presented in lecture performs the bulk copy or initialization of a page (or any arbitrary sized memory region) completely within DRAM, with support from the memory controller.

Suppose we have a cache-coherent system with six cores, a 32 KB L1 cache in each core, a 1MB private L2 cache per core, a 16MB shared L3 cache across all cores, and a single shared DRAM controller across all cores. The system supports RowClone as described in class. Physical page size is 8KB. MESI protocol is employed to keep the caches coherent.

Suppose Core 1 sends a request to the memory controller to Copy Physical Page 10 to Physical Page 12 via RowClone. Assume the two pages reside in the same DRAM subarray and the copy can be done with two consecutive ACTivate commands as a result.

(a) To maintain correctness of data in such a system, what should the memory controller do before performing the RowClone request? Explain step by step. Be precise.

Step 1:

Step 2:

(b) How much data transfer is eliminated from the main memory data bus with this particular copy operation? Justify your answer and state your assumptions.

Amount of data eliminated:

Justification:

---

[1]Seshadri et al., *"RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization."* In Proceedings of the 46th International Symposium on Microarchitecture (MICRO), 2013.

3. Tiered-difficulty [150 points]

Recall from your required reading on Tiered-Latency DRAM that there is a near and far segment, each containing some number of rows. Assume a very simplified memory model where there is just one bank and there are two rows in the near segment and four rows in the far segment. The time to activate and precharge a row is 25ns in the near segment and 50ns in the far segment. The time from start of activation to reading data is 10ns in the near segment and 15ns in the far segment. All other timings are negligible for this problem. Given the following memory request stream, determine the optimal assignment (minimize average latency of requests) of rows in the near and far segment (assume a fixed mapping where rows cannot migrate, a closed-row policy, and the far segment is inclusive).

```
time 0ns   : row 0 read
time 10ns  : row 1 read
time 100ns : row 2 read
time 105ns : row 1 read
time 200ns : row 3 read
time 300ns : row 1 read
```

(a) What rows would you place in near segment? Hint: draw a timeline.

(b) What rows would you place in far segment?

(c) In 15 words or less, describe the insight in your mapping?

(d) Assume now that the mapping is dynamic. What are the tradeoffs of an exclusive design vs. an inclusive design? Name one advantage and one disadvantage for each.

(e) Assume now that there are eight (8) rows in the near segment. Below is a plot showing the number of misses to the near segment for three applications (A, B, and C) when run alone with the specified number of rows allocated to the application in the near segment. This is similar to the plots you saw in your Utility-Based Cache Partitioning reading except for TL-DRAM instead of a cache. Determine the optimal static partitioning of the near segment when all three of these applications are run together on the system. In other words, how many rows would you allocate for each application? Hint: this should sum to eight. Optimal for this problem is defined as minimizing total misses across all applications.
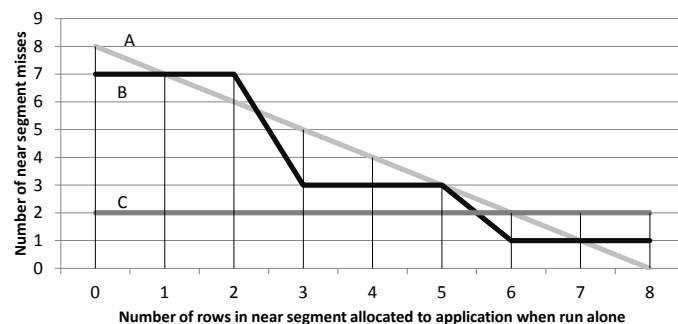
(1) How many near segment rows would you allocate to A?

(2) How many near segment rows would you allocate to B?

(3) How many near segment rows would you allocate to C?

4. DRAM Scheduling and Latency [150 points]

You would like to understand the configuration of the DRAM subsystem of a computer using reverse engineering techniques. Your current knowledge of the particular DRAM subsystem is limited to the following information:

- The physical memory address is 16 bits.

- The DRAM subsystem consists of a single channel, 2 banks, and 64 rows per bank.

- The DRAM is byte-addressable.

- The most-significant bit of the physical memory address determines the bank. The following 6 bits of the physical address determine the row.

- The DRAM command bus operates at 1 GHz frequency.

- The memory controller issues commands to the DRAM in such a way that *no command* for servicing a *later* request is issued before issuing a READ command for the current request, which is the oldest request in the request buffer. For example, if there are requests A and B in the request buffer, where A is the older request and the two requests are to different banks, the memory controller does *not* issue an ACTIVATE command to the bank that B is going to access *before* issuing a READ command to the bank that A is accessing.

- The memory controller services requests in order with respect to each bank. In other words, for a given bank, the memory controller first services the oldest request in the *request buffer* that targets the same bank. If all banks are ready to service a request, the memory controller first services the oldest request in the request buffer.

You realize that you can observe the memory requests that are waiting to be serviced in the request buffer. At a particular point in time, you take the snapshot of the request buffer and you observe the following requests in the request buffer (in descending order of request age, where the oldest request is on the top):

```
        Read 0xD780
 time   Read 0x280C
        Read 0xE4D0
        Read 0x2838
```

At the same time you take the snapshot of the request buffer, you start probing the DRAM command bus. You observe the DRAM command type and the cycle (relative to the first command) at which the command is seen on the DRAM command bus. The following are the DRAM commands you observe on the DRAM bus while the requests above are serviced.

```
Cycle 0  --- READ
Cycle 1  --- PRECHARGE
Cycle 8  --- PRECHARGE
Cycle 13 --- ACTIVATE
Cycle 18 --- READ
Cycle 20 --- ACTIVATE
Cycle 22 --- READ
Cycle 25 --- READ
```

Answer the following questions using the information provided above.

(a) What are the following DRAM timing parameters used by the memory controller, in terms of nanoseconds? If there is not enough information to infer the value of a timing parameter, write *unknown*.

   i) ACTIVATE-to-READ latency:

   ii) ACTIVATE-to-PRECHARGE latency:

   iii) PRECHARGE-to-ACTIVATE latency:

   iv) READ-to-PRECHARGE latency:

   v) READ-to-READ latency:

(b) What is the status of the banks *prior* to the execution of any of the above requests? In other words, which rows from which banks were open immediately prior to issuing the DRAM commands listed above? Fill in the table below indicating whether a bank has an open row, and if there is an open row, specify its address. If there is not enough information to infer the open row address, write *unknown*.

	Open or Closed?	Open Row Address
Bank 0		
Bank 1		

(c) To improve performance, you decide to implement the idea of Tiered-Latency DRAM (TL-DRAM) in the DRAM chip. Assume that a bank consists of a single subarray. With TL-DRAM, an entire bank is divided into a near segment and far segment. When accessing a row in the near segment, the ACTIVATE-to-READ latency *reduces* by 1 cycle and the ACTIVATE-to-PRECHARGE latency reduces by 3 cycles. When precharging a row in the near segment, the PRECHARGE-to-ACTIVATE latency reduces by 3 cycles. When accessing a row in the far segment, the ACTIVATE-to-READ latency *increases* by 1 cycle and the ACTIVATE-to-PRECHARGE latency increases by 2 cycles. When precharging a row in the far segment, the PRECHARGE-to-ACTIVATE latency increases by 2 cycles. The following table summarizes the changes in the affected latency parameters.

Timing Parameter	Near Segment Latency	Far Segment Latency
ACTIVATE-to-READ	-1	$+1$
ACTIVATE-to-PRECHARGE	-3	$+2$
PRECHARGE-to-ACTIVATE	-3	$+2$

Assume that the rows in the near segment have smaller row ids compared to the rows in the far segment. In other words, physical memory row addresses 0 through $N-1$ are the near-segment rows, and physical memory row addresses N through 63 are the far-segment rows.

If the above DRAM commands are issued 2 cycles faster with TL-DRAM compared to the baseline (the last command is issued in cycle 23), how many rows are in the near segment, i.e., what is N? Show your work.