CS 6501 - HW3 - DRAM

Huy Nguyen - mpc5ya

March 3, 2025

Q5 and Q6

I did the simulations on the portal and not sure how to show this, I wonder if I need to submit the traces and configs to get credits for those questions.

Q8. Analysis of DRAM Configurations for Random Access Pattern

Configuration	**Avg. BW**	**Avg. Power**	**Total Energy**	**Avg. Read Latency**
GDDR6_8Gb_x16	29.52	1485.80	1.49×10^8	1690.53
LPDDR4_8Gb_x16_2400	15.38	2113.22	2.11×10^8	200.48
HBM2_8Gb_x128 (total)	6.45	808.13	8.07×10^7	948.20
DDR4_8Gb_x8_3200	10.15	2248.97	2.25×10^8	129.75

Table 1: Random-Access Performance Metrics for Four DRAM Configurations (HBM2 values are summed across 8 channels)

a. Average Bandwidth

Average bandwidth represents the data transfer capability of the memory system. In our simulation, the **GDDR6_8Gb_x16** configuration leads with an average bandwidth of approximately 29.52 GB/s, reflecting its design for high-frequency, high-throughput operations typical in graphics and GPU-based applications. The **LPDDR4_8Gb_x16_2400** and **DDR4_8Gb_x8_3200** configurations offer moderate bandwidth values of 15.38 GB/s and 10.15 GB/s respectively. In contrast, even after correctly summing the bandwidth across all 8 channels, **HBM2_8Gb_x128** shows a total bandwidth of only about 6.45 GB/s under random access. This relatively low value suggests that the provided memory traces are not issuing enough concurrent requests to fully saturate HBM's potential—even after increasing the number of requests (e.g., using `-n 100000+`), the access pattern remains the limiting factor.

b. Average Power

Average power consumption is critical for energy-sensitive systems. The simulation shows that the **HBM2_8Gb_x128** configuration consumes only about 808.13 mW, which is significantly lower than the power consumption of LPDDR4 (2113.22 mW) and DDR4 (2248.97 mW). The **GDDR6_8Gb_x16** configuration is intermediate at approximately 1485.80 mW. This lower power draw reinforces HBM2's potential for use in environments where energy efficiency is paramount.

c. Total Energy

Total energy consumption, measured in picojoules (pJ), reflects both power usage and operational duration. The **GDDR6_8Gb_x16** configuration consumes about 1.49×10^8 pJ, while both **LPDDR4_8Gb_x16_2400** and **DDR4_8Gb_x8_3200** consume higher amounts (approximately 2.11×10^8 pJ and 2.25×10^8 pJ, respectively). In contrast, the **HBM2_8Gb_x128** configuration uses only about 8.07×10^7 pJ, highlighting its superior energy efficiency under the simulated random-access workload.

d. Average Read Latency

Average read latency, expressed in clock cycles, indicates the responsiveness of the memory system. The **GDDR6_8Gb_x16** configuration exhibits the highest read latency at around 1690.53 cycles, likely due to its high clock frequency where each cycle is very short. The **HBM2_8Gb_x128** configuration shows an average read latency of approximately 948.20 cycles. Meanwhile, the **LPDDR4_8Gb_x16_2400** and **DDR4_8Gb_x8_3200** configurations have much lower latencies at about 200.48 cycles and 129.75 cycles respectively. Thus, while HBM2 is very energy efficient, its latency under random-access conditions is moderately high compared to DDR4 and LPDDR4.

Observations and Insights

- **Bandwidth vs. Access Pattern:**
  - **GDDR6** offers the highest bandwidth (29.52 GB/s), making it ideal for high-throughput applications.
  - **LPDDR4** and **DDR4** provide moderate bandwidths.
  - **HBM2** (even when summing across 8 channels) delivers only about 6.45 GB/s, suggesting that the memory traces are not generating enough concurrent activity to leverage HBM's full potential.
- **Power Consumption:**
  - **HBM2** is the most energy efficient with the lowest average power (808.13 mW).
  - **GDDR6** has moderate power usage, while **LPDDR4** and **DDR4** consume significantly more power under the tested workload.
- **Energy Efficiency:**
  - **HBM2** also excels in total energy consumption, using only 8.07×10^7 pJ compared to over 1.49×10^8 pJ for GDDR6 and over 2×10^8 pJ for the other configurations.
- **Latency Considerations:**
  - **GDDR6** exhibits the highest cycle count for read latency (1690.53 cycles), although its high frequency may mitigate the real-time delay.
  - **HBM2** shows a moderate read latency (948.20 cycles) which is higher than that of LPDDR4 (200.48 cycles) and DDR4 (129.75 cycles) under random access.
- **Trade-Offs:**
  - High-bandwidth designs like GDDR6 may achieve superior throughput but at the expense of higher latency and moderate power consumption.
  - Energy-efficient designs like HBM2 consume much less power and energy but may not reach their theoretical peak bandwidth if the access pattern does not generate sufficient concurrent activity.
  - LPDDR4 and DDR4 strike a balance between throughput, power, and latency, making them suitable for general-purpose applications.

Question 9

Data Overview

Table 2 provides the measured values of activation (`num_act_cmds`) and precharge (`num_pre_cmds`) commands for four DRAM configurations under three access patterns: *random*, *mix*, and *stream*. Note that for the HBM2_8Gb_x128 configuration, the originally reported values have now been multiplied by 8 (to account for all channels).

Access Pattern	Configuration	num_act_cmds	num_pre_cmds
mix	HBM2_8Gb_x128	5312	5224
mix	DDR4_8Gb_x8_3200	5317	5301
mix	GDDR6_8Gb_x16	5436	5421
mix	LPDDR4_8Gb_x16_2400	5435	5428
random	GDDR6_8Gb_x16	7549	7546
random	LPDDR4_8Gb_x16_2400	9992	9985
random	HBM2_8Gb_x128	9856	9744
random	DDR4_8Gb_x8_3200	9999	9983
stream	DDR4_8Gb_x8_3200	94	86
stream	LPDDR4_8Gb_x16_2400	121	119
stream	GDDR6_8Gb_x16	38	36
stream	HBM2_8Gb_x128	536	512

Table 2: Activation (`ACT`) and Precharge (`PRE`) Command Counts for Random, Mixed, and Streaming Access Patterns

Observations

Random Access

- **High Command Counts:** Under random accesses, all configurations exhibit high command counts. For instance, DDR4_8Gb_x8_3200 and LPDDR4_8Gb_x16_2400 both issue around 9999 and 9992 activations respectively, while GDDR6_8Gb_x16 issues about 7549. With the correction, HBM2_8Gb_x128 now issues approximately 9856 activations and 9744 precharges. This indicates that, when considering all channels, HBM2 experiences activation and precharge overheads similar to LPDDR4 and DDR4 under random access.
- **Frequent Row Switching:** The high command counts arise because random accesses frequently target different rows, requiring the controller to continuously open new rows and precharge the old ones.

Mixed Access

- **Moderate Command Counts:** The mixed pattern results in command counts that are intermediate between random and streaming accesses. DDR4, GDDR6, and LPDDR4 exhibit activations in the range of 5317 to 5436, while HBM2, after multiplication, now shows around 5312 activations and 5224 precharges—very similar to the other configurations. This suggests that the mixed trace allows for some row reuse, reducing the frequency of activation and precharge commands compared to pure random access.
- **Interleaving of Random and Sequential Access:** The mixed access pattern combines segments with high row locality and segments with random jumps, resulting in a moderate overall command count.

Streaming Access

- **Low Command Counts for Most Configurations:** Streaming accesses result in very low command counts for DDR4 (94 activations), LPDDR4 (121 activations), and GDDR6 (38 activations) because sequential access promotes row buffer reuse.
- **Higher Aggregate Commands for HBM2:** In contrast, HBM2_8Gb_x128, when corrected to account for all 8 channels, now has a total of 536 activations and 512 precharges. Although this might seem high compared to the others, it reflects the fact that HBM2 has multiple independent channels; each channel might issue only a few commands (on average about 67 activations per channel), but when summed across all channels, the total is higher.

Explanation for the Large Differences

The differences in the number of activation (`ACT`) and precharge (`PRE`) commands are primarily driven by the degree of row buffer locality. In random access patterns, there is minimal row buffer locality; nearly every memory request targets a different row, which forces the memory controller to activate a new row and precharge the previous one for each access. This lack of reuse results in high command counts. In contrast, streaming access patterns exhibit high row buffer locality, allowing multiple consecutive accesses to be served from the same row. As a result, once a row is activated, many requests can be processed without needing additional activations or precharges, thereby significantly reducing the command overhead. Mixed access patterns, which combine elements of both random and sequential accesses, show intermediate behavior where some requests benefit from row reuse while others do not.

Additionally, the memory controller's scheduling policy plays an important role in this behavior. Under random access, the controller rarely encounters consecutive requests for the same row, so it must frequently issue new activation and precharge commands. However, in streaming access, the controller can keep a row active for a longer duration, minimizing the number of commands needed. Together, these factors—row buffer locality and memory controller scheduling—explain the observed variations in command counts across different access patterns.

In summary, when the HBM2 data is correctly scaled to account for its 8 channels, its activation and precharge command counts in random and mixed access patterns are comparable to those of DDR4, LPDDR4, and GDDR6. In the streaming pattern, while HBM2's total command count is higher, this is a natural consequence of summing across multiple channels (each of which individually issues only a few commands). These results underscore the impact of access patterns on DRAM efficiency and highlight how the benefits of row buffer locality and controller scheduling vary across different DRAM architectures.

Question 10

What kind of DRAM would you use for the following cases and why?

1. A high-throughput system
2. A power-constrained system

Table 3 shows representative average bandwidth and average power values for four DRAM configurations under different access patterns. Note that for the HBM2_8Gb_x128 configuration, the values have been multiplied by 8 to reflect the aggregate performance across all channels.

Configuration	**Average Bandwidth (GB/s)**	**Average Power (mW)**
GDDR6_8Gb_x16 (mix)	38.77	1350.60
GDDR6_8Gb_x16 (random)	29.52	1485.80
GDDR6_8Gb_x16 (stream)	38.78	882.04
HBM2_8Gb_x128 (mix)	6.45	808.13
HBM2_8Gb_x128 (random)	6.45	808.13
HBM2_8Gb_x128 (stream)	6.56	730.56
LPDDR4_8Gb_x16_2400 (mix)	15.39	1731.96
LPDDR4_8Gb_x16_2400 (random)	15.38	2113.22
LPDDR4_8Gb_x16_2400 (stream)	15.38	1269.83
DDR4_8Gb_x8_3200 (mix)	10.15	1934.44
DDR4_8Gb_x8_3200 (random)	10.15	2248.97
DDR4_8Gb_x8_3200 (stream)	10.16	1534.90

Table 3: Revised Representative Average Bandwidth and Power for Different DRAM Configurations (HBM2 values multiplied by 8)

(a) High-Throughput System

For a high-throughput system, the key metric is the maximum achievable bandwidth. According to the simulation data, the **GDDR6_8Gb_x16** configuration consistently delivers the highest average bandwidth, ranging from approximately 29.5 GB/s (random) to nearly 38.8 GB/s (mixed and streaming). In comparison, even after accounting for all channels, the aggregate bandwidth of **HBM2_8Gb_x128** is only around 6.45–6.56 GB/s, while **LPDDR4_8Gb_x16_2400** and **DDR4_8Gb_x8_3200** achieve moderate bandwidths of about 15.38 GB/s and 10.15 GB/s, respectively.

- GDDR6 is engineered for high-frequency, high-throughput applications such as graphics rendering and GPU computing.
- Its architecture and interface allow it to achieve very high data rates, which is crucial for systems where peak data transfer performance is essential.

Assumptions and Notes:

- Although HBM2 is designed for high bandwidth, the simulation traces used do not generate enough concurrent requests to fully utilize its potential.
- The reported simulation results for GDDR6 indicate that under these particular workloads, GDDR6 outperforms other DRAM types in terms of bandwidth.

(b) Power-Constrained System

For systems where power and thermal budgets are limited, the focus shifts to minimizing power consumption and energy usage. The simulation data shows that the **HBM2_8Gb_x128** configuration has the lowest average power consumption, with values around 808 mW (mix and random) and approximately 730 mW (stream). This is significantly lower than the power figures for GDDR6 (882 to 1485 mW), LPDDR4 (1269 to 2113 mW), and DDR4 (1535 to 2249 mW).

- HBM2 uses a 3D-stacked architecture with a wide interface, which allows it to operate at lower clock speeds and achieve better power efficiency.

- Even though the simulation shows modest bandwidth for HBM2 under the tested random/mixed workloads, its low power draw and total energy consumption make it ideal for power-constrained applications.

Assumptions and Notes:

- While LPDDR4 is generally marketed as a low-power option, under heavy or continuous random/mixed workloads it can still draw significant power.
- In real-world scenarios, additional factors such as cost, packaging, and system integration will also influence the choice. However, for strictly power-constrained environments, the low power consumption of HBM2 is a clear advantage.

Conclusion

- **High Throughput System:** Based on the provided simulation metrics, **GDDR6_8Gb_x16** is the top choice due to its superior average bandwidth. It is ideal for workloads that require maximum data transfer rates, such as high-performance graphics or compute-intensive applications, even though it comes with relatively higher power consumption.
- **Power-Constrained System:** For systems where power efficiency is critical, **HBM2_8Gb_x128** is the most suitable option. Despite its lower measured bandwidth under the given trace conditions, its substantially lower power consumption and overall energy usage make it a better choice in environments with strict power or thermal limits.

Question 11

Figure 1 shows a histogram of read latencies for the `DDR4_8Gb_x8_3200` configuration under a mixed workload. The majority of reads complete around or below 100–150 cycles, yet a smaller portion extends to higher latencies (with a 99$^{\text{th}}$ percentile at 724 cycles and a maximum of 909 cycles). Figure 2 similarly illustrates a broad distribution for write latencies, averaging 387 cycles but occasionally reaching over 1000 cycles (max 1726). Figure 3 shows that interarrival times between requests hover around 10 cycles, suggesting a steady but not overly dense request stream.

Analysis of Latency Variability

DRAM latency variability is primarily due to a combination of dynamic factors. One key factor is the difference between row buffer hits and misses: when a request targets an already open row (a row buffer hit), it is serviced quickly, whereas a request that accesses a different row must incur the overhead of an activate (ACT) command and often a precharge (PRE) command to close the previous row, thereby significantly increasing the latency. Additionally, bank conflicts and the memory controller's scheduling policies play a major role. If multiple requests contend for the same bank, the controller must queue them, and the specific scheduling algorithm (such as FR-FCFS or open/closed page policies) further determines the order in which requests are processed, leading to variable delays. Furthermore, the inherent timing constraints of DDR4—such as tRCD, tCL, tRP, and tWR—mean that the exact sequence of commands (activate, read, write, precharge, and even refresh operations) can cause individual requests to experience different delays. Read and write interactions also contribute: many controllers prioritize reads over writes by buffering write requests until an idle period or until the write buffer fills, which can result in higher write latencies (for example, the average write latency of 387 cycles can sometimes exceed 1200 cycles). Finally, periodic refresh operations temporarily block DRAM operations, so if a request occurs just before a refresh, it may be delayed until the refresh completes, leading to sporadic spikes in latency.

Conclusion

These histograms confirm that real-world DRAM latencies are not uniform but depend on the current state of the memory system, ongoing requests, and the memory controller's scheduling policies. Although many requests complete quickly (leading to a peak in the histogram at lower latencies), the tail of the distribution (up to hundreds or even over a thousand cycles) reveals that some requests inevitably experience additional delays due to row misses, bank conflicts, and other timing constraints.

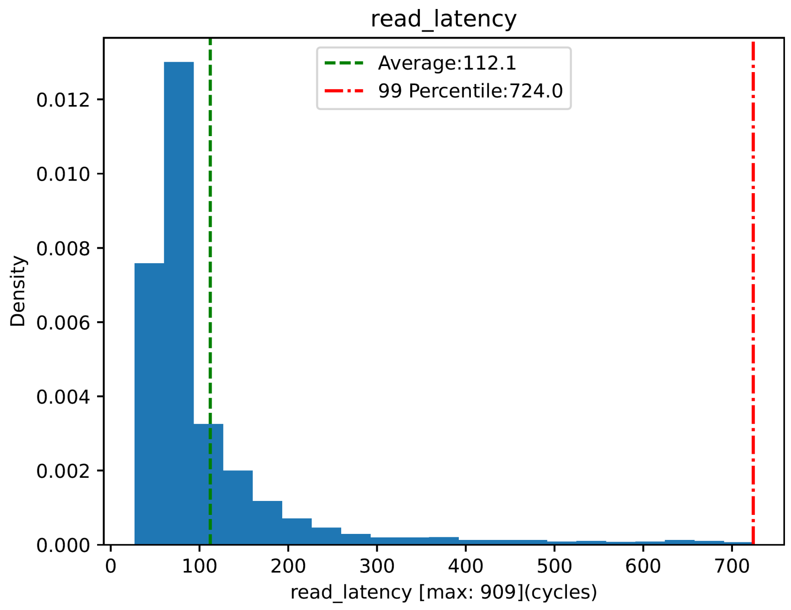

Figure 1: Read Latency Distribution: Average 112.1 cycles, 99th Percentile 724 cycles, Max 909 cycles.

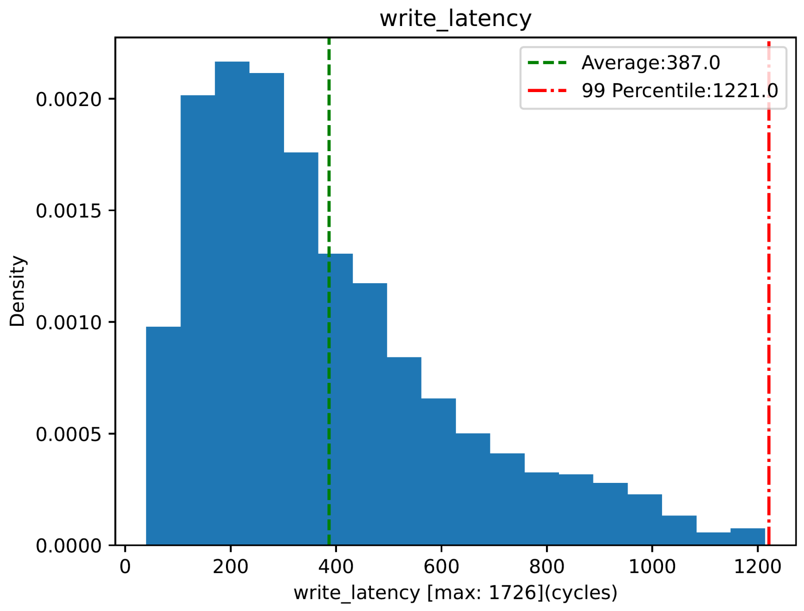

Figure 2: Write Latency Distribution: Average 387.0 cycles, 99th Percentile 1221 cycles, Max 1726 cycles.

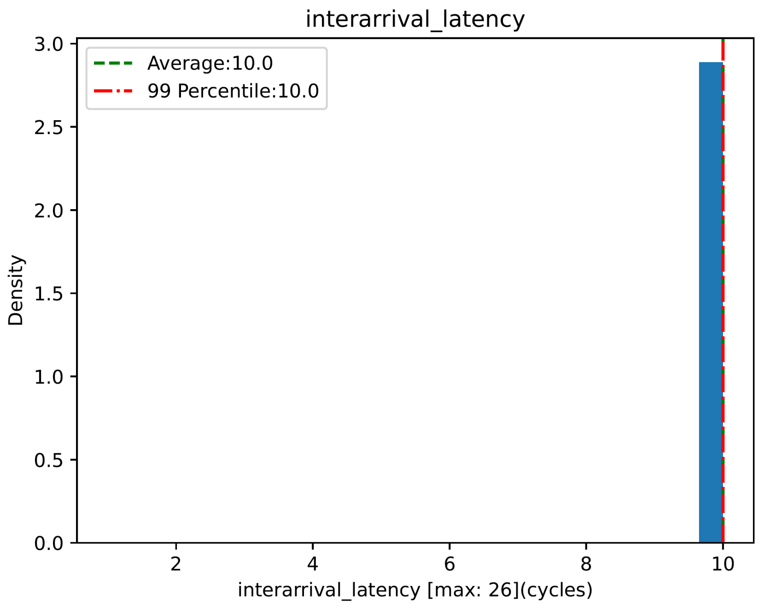

Figure 3: Interarrival Latency Distribution: Average 10.0 cycles, 99th Percentile 10.0 cycles, Max 26 cycles.