Homework Assignment 6 solution

CS/ECE 6810: Computer Architecture
April 11, 2018

Vikas Rao - U1072596

April 18, 2018

1. **Memory Scheduling Mechanisms (20 points):** For the following memory access pattern, estimate when each memory access completes for two different scheduling mechanisms: **open-page policy** and **close-page policy**. You are allowed to reorder requests already waiting in the memory controller. The access pattern only specifies the row being touched. All accesses are to the **same** bank. Assume that bus latencies are zero. Assume that the bank is already pre-charged at time 0. Assume that pre-charge takes 20 ns, loading a row buffer takes 20 ns, and cache line transfer to output pins also takes 20 ns.

   In the table below 1 we are assuming that only after a row buffer data is read into memory channel, we can start pre-charging and service the next request. In case of a buffer hit, i.e. current row requested is same as the previous row, the open page policy will just incur 20ns penalty for cache line transfer to output pins. In cases of a conflict, i.e. a different row requested than what is in the row buffer, the penalty is - Precharge(PRE) + row buffer read (ACT) + cache line read (RD) = 60ns. None of the row accesses in the open page policy benefits from the FR-FCFS scheduling.

   In the closed-page policy, we can benefit from using FR-FCFS scheduling by queuing the third X read before the first Y read request.

   Table 1: original ordering

Row being accessed	**Arrival time at memory controller**	**Open-page**	**Close-page**
X	10 ns	50ns	50ns
X	70 ns	90ns	110ns
Y	90 ns	150ns	190ns
X	100 ns	210ns	130ns
Y	180 ns	270ns	210ns

2. **Memory System Design (20 points):** Modern systems have processors with four memory channels. Assume that a memory channel has 64 bits for data and 24 bits for address and command. Assume that a memory channel is connected to eight x8 memory chips. Consider a new memory system that shrinks the data bus of the memory channel to 32 bits. Assume that such a channel would be connected to four x8 memory chips.

Mention **two pros** and **two cons** of the new memory system (Be very concise).

Assumptions - The row buffer size is kept same in both the cases and the frequency of operations are also kept same.

Pros -

1. Is more energy efficient since the number of chips accessed for every request are less as compared to the original system.

2. reduced latency for refresh of storage since there are less chips per rank.

3. Channel pin count is decreased

Cons -

1. Latency for cache line accesses as the average 64B cache line read will take multiple cycles (Decreased memory bandwidth).

2. Reduced capacity for DRAM storage since the number of chips per rank are less as compared to the original system.

3. Doesn't explore the spatial locality concept since there are reduced number of chips per bank read into row buffer.

3. **Virtually Indexed Physically Tagged Cache (20 points):** Assume that the OS uses a minimum page size of 4 KB. Assume that your L1 cache must be 8-way set-associative. If you're trying to correctly implement a virtually indexed physically tagged cache (with no additional support from the OS), what is the largest L1 cache that you can design?

   Page size is $4KB = 2^{12}$

   Therefore, a virtually indexed physically tagged cache can use a maximum of 12 index bits to avoid aliasing issues without extra support.

   Now, as the cache is 8 way set associative, the maximum size of the cache can be $4*8KB = 32KB$.

4. **DRAM Control (10 points):** Express the latency of a memory load as a function of tRAS, tRP, tRCD, tCL in the following situations:

   i The correct row is already placed in the row buffer

   ii Another row is already placed in the row buffer.

   Assuming the following,

   tRAS = row active strobe, tRP = row precharge, tRCD = row to column delay, tCL = column access strobe.

   a) The correct row is already placed in the row buffer: (Row hit), therefore tCL.

   b) Another row is already placed in the row buffer: (Row miss), therefore tRP + tRCD+ tCL

5. **DRAM Control Tasks Request Scheduling (20 points):** Find the total number of commands sent by an in-order command scheduler for the given sequence of memory addresses for reads, using the following address mapping scheme:

   Address = row(12):bank(3):rank(1):channel(0):column(16)

   Assume that all banks are initially precharged.

Table 2: commands

Address	**Rank**	**Bank**	**Row**	**Col**	**PRE**	**ACT**	**RD**
00040108	0	2	000	0108		*	*
01040101	0	2	010	0101	*	*	*
00161804	0	3	001	1804		*	*
01040104	0	2	010	0104			*

As seen from the decoding and table 3, the in-order command scheduler will send 8 commands for these sequence of memory addresses.