CS4100: Computer System Design

Memory Hierarchy Design

Madhu Mutyam
PACE Laboratory
Department of Computer Science and Engineering
Indian Institute of Technology Madras

Sept 3-7, 2015

Example #1

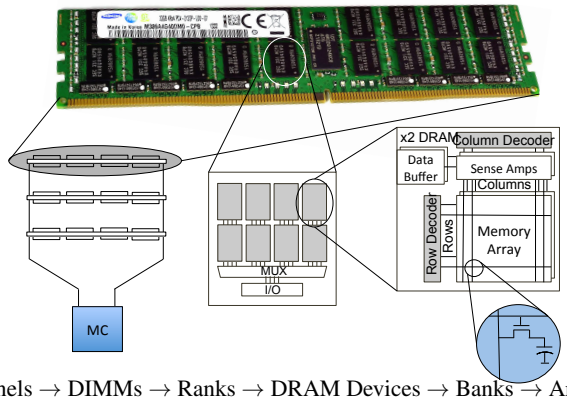
- Consider three levels of cache hierarchy
  - L1 cache: 2-cycle latency with 95% hit rate
  - L2 cache: 14-cycle latency with 70% hit rate
  - L3 cache: 30-cycle latency with 60% hit rate
  - Main memory: 250-cycle latency

$$\begin{aligned}\text{Average Memory Access Time (AMAT)} &= 2 \times 0.95 + \\ &\quad 14 \times 0.05 \times 0.7 + \\ &\quad 30 \times 0.05 \times 0.3 \times 0.6 + \\ &\quad 250 \times 0.05 \times 0.3 \times 0.4 \\ &= 1.9 + 0.49 + 0.27 + \mathbf{1.5} \\ &= 4.16\end{aligned}$$

36% of AMAT is contributed by the main memory!!!

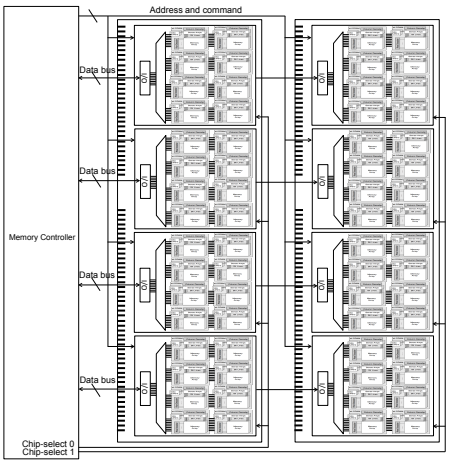
32GB DRAM DIMM

DRAM-based Main Memory Organization

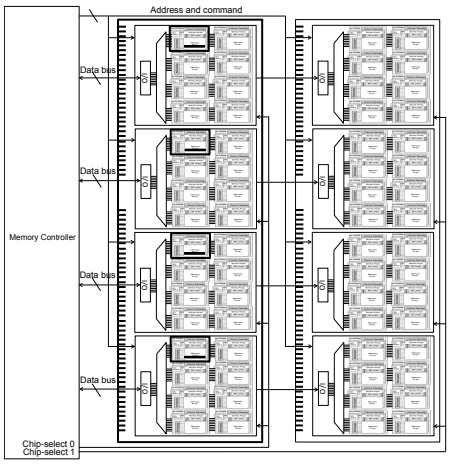


Channels → DIMMs → Ranks → DRAM Devices → Banks → Array of rows and columns

2-Rank 4-Device 8-Bank DIMM-DRAM Organization



Accessing a DRAM Row (Page)



Why Do We Need Such an Organization?

- Provides bandwidth through pipelining requests or parallel access
- Pipelining request processing
  - concurrency at rank-level and bank-level
- Achieving parallel access
  - Multiple DRAM devices act in unison at the rank level
  - Multiple DRAM arrays act in unison at the bank level

Example #2

Consider a 2-rank DIMM with each rank having four 4Gb devices in it and each device implements eight x8 sub-banks. What are the capacities of the DIMM, the rank, and the bank? What is the size of a row, if a bank has 64K rows in it?

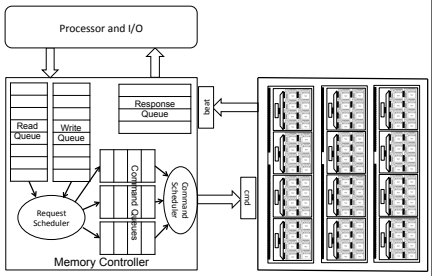
- Since each rank has four 4Gb devices, the capacity of each rank is (4×4)Gb = 16Gb or 2GB.
- As the DIMM has two ranks in it, the capacity of the DIMM is 4GB.
- As each device implements 8 sub-banks and there are four devices, the total number of banks in a rank is 8, where a bank is equally distributed among all the four devices.
- Thus, the capacity of each bank is $(16/8)$Gb = 2Gb or 256MB.
- As the bank capacity is 256MB and the total number of rows in a bank is 64K, the row size is $(256\text{MB}/64\text{K})$ = 4KB.

Basic DRAM Operations

- Address lines are multiplexed to reduce the number of pins
  - Send row address when *Row Address Strobe* (RAS) is asserted
  - Send column address when *Column Address Strobe* (CAS) is asserted
- To access data
  - *Precharge* (PRE) – places the bank in a voltage state that is suitable for activation
  - *Activate* (ACT) – reads an entire row of bits into the sense amplifiers
  - *Column Access* (Rd/Wr) – transfers data to/from the memory controller
- To retain data
  - *Restore* – DRAM reads are self-destructive
  - *Refresh* (REF) – DRAM cells lose charge due to leakage
    - Periodic charge/refill is needed
    - Retention time: $64ms$ for $< 85^oC$ temperature and $32ms$ for $85^oC - 95^oC$ temperature

DRAM Memory Controller

- Functions:
  - Converts memory requests to DRAM commands
  - Enforces timing constraints
  - Row-buffer management policies
  - Address mapping
  - Refresh management
  - Memory scheduling



- Microarchitecture of a DRAM memory controller determines the latency and sustainable bandwidth
- Design goals of a DRAM memory controller are:
  - Maximize system performance
  - Minimize power consumption

Converting Memory Requests to Memory Commands

- Memory requests: *Load* and *Store*
- Memory commands: *PRE*, *ACT*, and *CAS*

S.No	Memory Request
R1	LOAD from Row 0 Column 4
R2	LOAD from Row 0 Column 5
R3	LOAD from Row 1 Column 1
R4	STORE to Row 1 Column 3

Request Order: R1 → R2 → R3 → R4

Memory Request	Memory Commands
LOAD from Row 0 Column 4	PRE → ACT → CAS
LOAD from Row 0 Column 5	CAS
LOAD from Row 1 Column 1	PRE → ACT → CAS
STORE to Row 1 Column 3	CAS

Exploits page access locality

Request Order: R1 → R2 → R4 → R3

Memory Request	Memory Commands
LOAD from Row 0 Column 4	PRE → ACT → CAS
LOAD from Row 0 Column 5	CAS
STORE to Row 1 Column 3	PRE → ACT → CAS
LOAD from Row 1 Column 1	CAS

Exploits page access locality, but incurs store-to-load latency

Request Order: R1 → R4 → R3 → R2

Memory Request	Memory Commands
LOAD from Row 0 Column 4	PRE → ACT → CAS
STORE to Row 1 Column 3	PRE → ACT → CAS
LOAD from Row 1 Column 1	CAS
LOAD from Row 0 Column 5	PRE → ACT → CAS

Exploits page access locality partially

Request Order: R1 → R4 → R2 → R3

Memory Request	Memory Commands
LOAD from Row 0 Column 4	PRE → ACT → CAS
STORE to Row 1 Column 3	PRE → ACT → CAS
LOAD from Row 0 Column 5	PRE → ACT → CAS
LOAD from Row 1 Column 1	PRE → ACT → CAS

Page access locality is not exploited

Few DRAM Timing Parameters

- *Row Precharge Delay* (t_{RP}):
  - The minimum time gap between a PRE and an ACT to the same bank
- *Row-to-Column Delay* (t_{RCD}):
  - The time interval between row access and data ready at sense amplifiers
- *Row Address Strobe* (t_{RAS}): t_{RCD} + Data restoration time
  - The minimum time before issuing a PRE to the same bank after an ACT
- *Column Address Strobe* (t_{CAS} or t_{CL}):
  - The time interval between column access command and the start of data return by the DRAM device
- *Column-to-Column Delay* (t_{CCD}):
  - The minimum column command timing, determined by internal burst (prefetch) length. Multiple internal bursts are used to form longer burst for column reads
- *Burst Length Delay* (t_{BL}):
  - The duration that data burst occupies on the data bus

Few DRAM Timing Parameters (Contd)

- *Write-to-Read Delay* (t_{WTR}):
  - The time that the I/O gating resources are released by the column-write
  - The other timing parameters to deal with column-write command are *column-write delay* (t_{CWD}) and *write recovery time* (t_{WR})
- *Row Cycle Time* (t_{RC}): $t_{RAS} + t_{RP}$
  - The time interval between accesses to different rows in a bank
- *Refresh Cycle Time* (t_{RFC}):
  - The time between a refresh and an ACT
- *Rank-to-Rank Switch Delay* (t_{RTRS}):
  - The time to switch from one rank to another rank
- Timing parameters for an 8Gb DDR3L/DDR4 DRAM device[1]:

Parameter	DRAM Cycles	Parameter	DRAM Cycles
t_{RCD}	11	t_{RAS}	28
t_{RP}	11	t_{CAS}	11
t_{RC}	39	t_{CCD}	4
t_{WTR}	6	t_{BL}	4
t_{RFC}	280	t_{RTRS}	2

[1] source: JEDEC, 2012

Row-Buffer Management Policies

- *Open-page* row-buffer policy
  - Read latency can be either t_{CAS} (on a *row-buffer hit*) or $t_{RP} + t_{RCD} + t_{CAS}$ (on a *row-buffer miss*)
  - Better for memory request sequences that show high access locality
- *Closed-page* row-buffer policy
  - Read latency is $t_{RCD} + t_{CAS}$
  - Better for memory request sequences with low access locality and/or low request rates
- *Hybrid* row-buffer policy
  - Use a timer to control the sense amplifiers
  - The timer is set to a predefined value when a row is activated or reaccessed
  - Precharge command is issued when the timer reaches zero
  - Better for memory request sequences whose request rate and access locality can change dynamically

Time	Memory Read Request	Page Status	Memory Commands Issued		
			Open-Page	Closed-Page	Hybrid (50)
0	Row 0 Column 4	Empty	ACT + CAS	ACT + CAS	ACT + CAS
40	Row 0 Column 5	Hit	CAS	ACT + CAS	CAS
70	Row 1 Column 3	Miss	PRE + ACT + CAS	ACT + CAS	PRE + ACT + CAS
140	Row 2 Column 1	Miss	PRE + ACT + CAS	ACT + CAS	ACT + CAS

Address Mapping

- It can affect the DRAM memory system performance
- Goal is to minimize the probability of bank conflicts in temporally adjacent requests and maximize the parallelism in the memory system
- Parallelism is available at various granularities:
  - Channels – the highest degree of parallelism
  - Ranks – rank-to-rank switching penalties in DDRx SDRAMs
  - Banks – if shared resources are available, consecutive requests can be sent to different banks of a given rank
  - Rows – only one row per bank can be active at any given instance in time
  - Columns – map the column address to the lower (higher) address bits of a given physical address in open (closed)-page system

Baseline Address Mapping Schemes

- Let the total size of the memory be $K \times L \times B \times R \times C \times V$, where
  - K: # of channels ($= 2^k$)
  - L: # of ranks per channels ($= 2^l$)
  - B: # of banks per rank ($= 2^b$)
  - R: # of rows per bank ($= 2^r$)
  - C: # of columns per row ($= 2^c$)
  - V: # of bytes per column ($= 2^v$)
- The number of bytes per row ($C \times V$) can be expressed as $N \times Z$, where
  - N: # of cache lines per row ($= 2^n$)
  - Z: # of bytes per cache line ($= 2^z$)
- The baseline open-page address mapping scheme is $r : l : b : n : k : z$
- The baseline closed-page address mapping scheme is $r : n : l : b : k : z$

		B_0	B_1	B_0	B_1	B_0	B_1	B_0	B_1
C_0	R_0	0	8	2	10	4	12	6	14
	R_1	16	24	18	26	20	28	22	30
C_1	R_0	1	9	3	11	5	13	7	15
	R_1	17	25	19	27	21	29	23	31

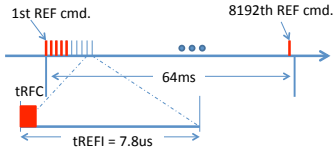
$(r : l : b : n : k : z)$

		B_0	B_1	B_0	B_1	B_0	B_1	B_0	B_1
C_0	R_0	0	2	8	10	16	18	24	26
	R_1	4	6	12	14	20	22	28	30
C_1	R_0	1	3	9	11	17	19	25	27
	R_1	5	7	13	15	21	23	29	31

$(r : n : l : b : k : z)$

Refresh Management

- Read data and restore it in DRAM devices, before the worst-case data decay time
- Consumes available bandwidth and power
- Goal: minimize controller complexity or bandwidth impact or power consumption
- *All-bank concurrent refresh*: a single refresh command to the DRAM device
  - *Refresh address register* to store the address of the last refreshed row
- Memory controller issues 8192 refresh commands to the DRAM device for every 64ms
- *Distributed refresh*:



Memory Scheduling

- Scheduling at request-level and command-level
- Two separate queues to hold reads and writes
  - Memory controller works in *read major mode* and *write major mode*
- Request scheduler can select requests in out-of-order from the request queues
- Per-bank command queues can be considered
- Command scheduler selects a command queue based on *Round-Robin*, *First Ready First Serve* (FRFS), or *age-based* policies
- Command scheduler picks the commands from command queue in FIFO order

Summary

- To deal with the memory wall problem, we need to come up with efficient memory hierarchy design
- Effective, but low-overhead, cache replacement policies are needed to utilize the cache space efficiently
- Efficient row-management techniques, address mapping techniques and memory scheduling policies improve overall memory performance
- DRAMSim (http://www.eng.umd.edu/∼blj/dramsim/) can be used to simulate DRAM-based memory

Thank You