中国科学院大学
University of Chinese Academy of Sciences

Loongson

高级计算机系统结构

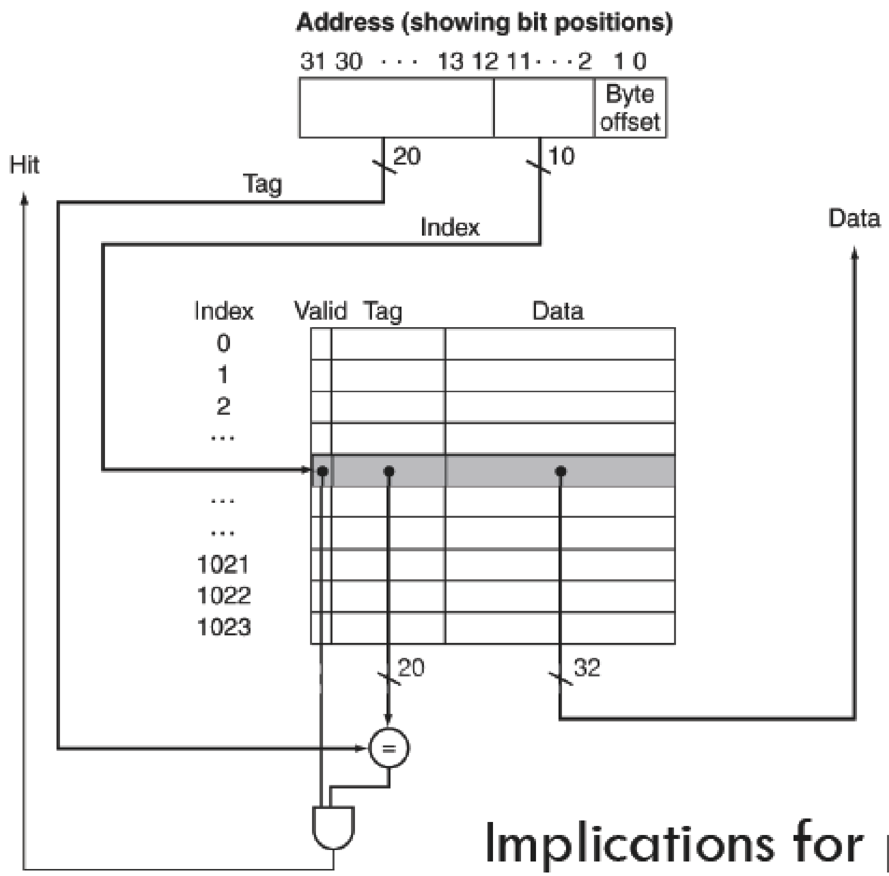
沈海华
shenhh@ucas.ac.cn

第四讲 存储系统设计：主存储器

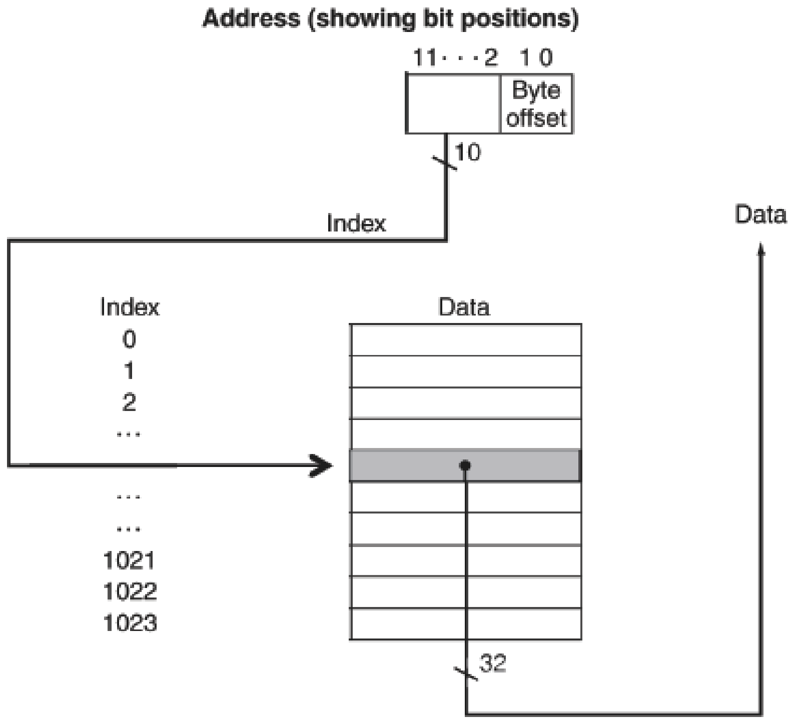
- 存储层次结构
  - 寄存器
  - Cache&TLB
  - Memory
  - Disk&Flash
  - 网络存储

Cache vs Local Store

- Cache



- Local Store



Implications for performance & power?

Local Stores: AMAT

AMAT = HitTime + MissRate * MissPenalty

- MissRate = 0%!

Consequences?

- Simpler performance analysis
- Less motivation for out-of-order cores

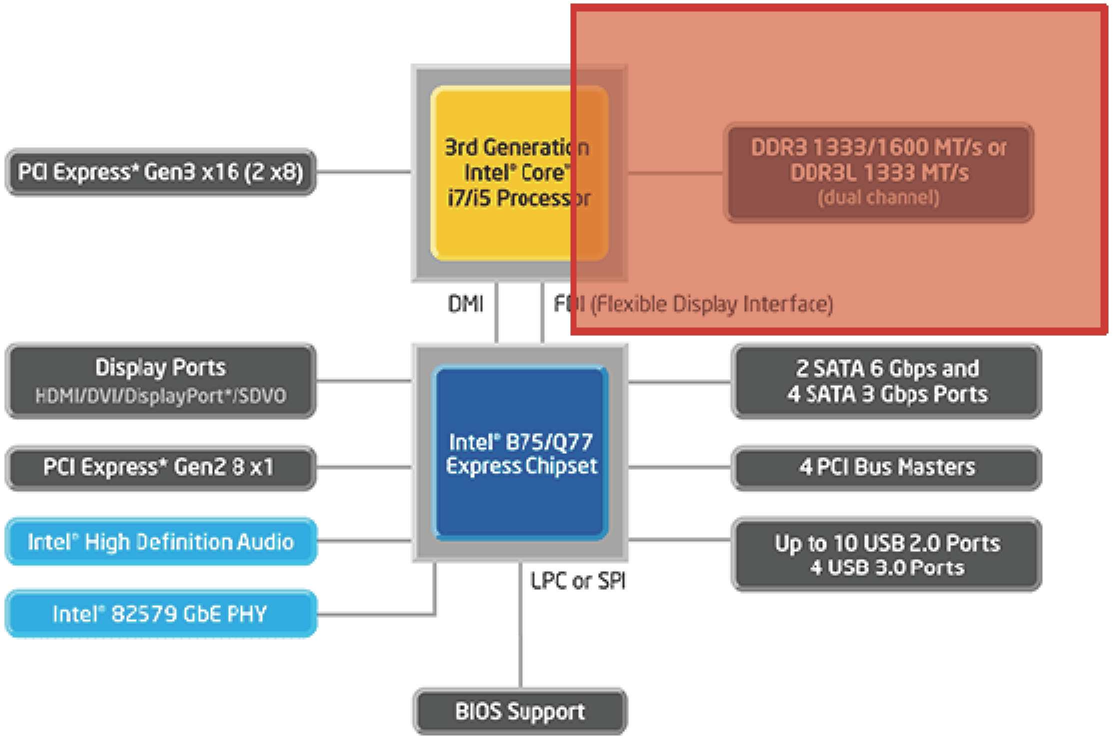
Local Stores: Operation

- LD/ST instructions to LS proceed normally
  - No LD/ST to non-LS memory
- DMA transfers (Direct Memory Access) to move data to/from main memory and LS
  - Bulk, like memcpy()
  - Asynchronous

```
dma(void *local_address, void *remote_address,
    int size, int tag, boolean direction);
```

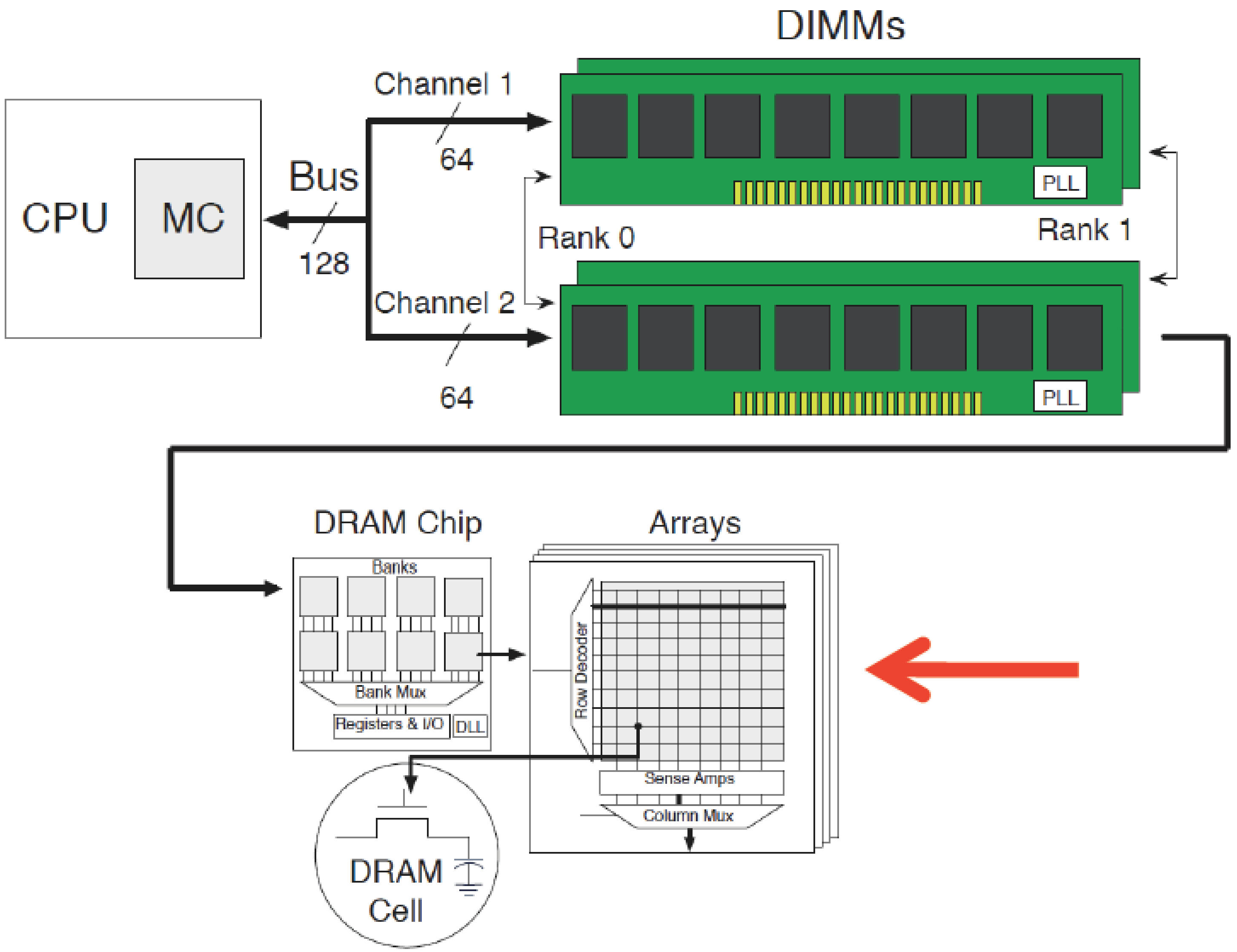
Today's lecture: Main Memory

- Main Memory
  - DRAM technology
  - DRAM as Main Memory
  - DRAM power considerations in a later lecture



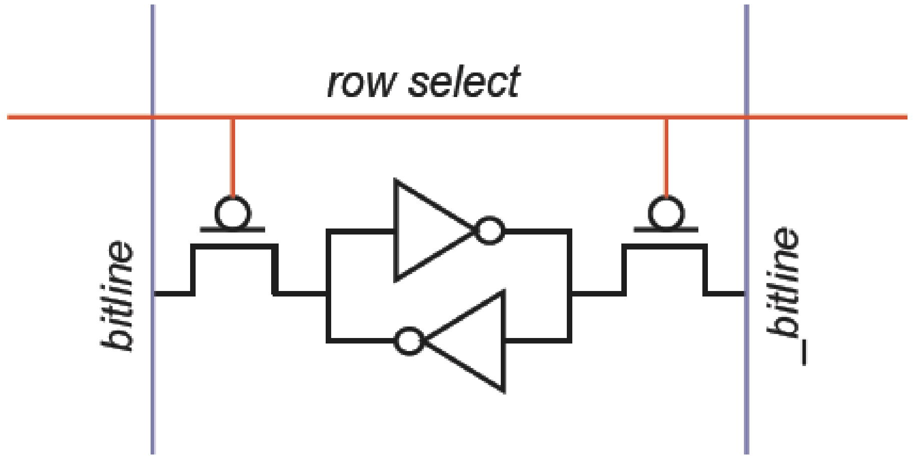
PCI Express* Gen3 x16 (2 x8)
3rd Generation Intel® Core™ i7/i5 Processor
DDR3 1333/1600 MT/s or DDR3L 1333 MT/s (dual channel)
DMI
FDI (Flexible Display Interface)
Display Ports HDMI/DVI/DisplayPort*/SDVO
PCI Express* Gen2 8 x1
Intel® High Definition Audio
Intel® 82579 GbE PHY
Intel® B75/Q77 Express Chipset
2 SATA 6 Gbps and 4 SATA 3 Gbps Ports
4 PCI Bus Masters
Up to 10 USB 2.0 Ports 4 USB 3.0 Ports
LPC or SPI
BIOS Support

Main Memory


DIMMs
Channel 1
64
CPU
MC
Bus
128
Rank 0
Rank 1
PLL
Channel 2
64
PLL
DRAM Chip
Arrays
Banks
Bank Mux
Registers & I/O
DLL
Row Decoder
Sense Amps
Column Mux
DRAM Cell

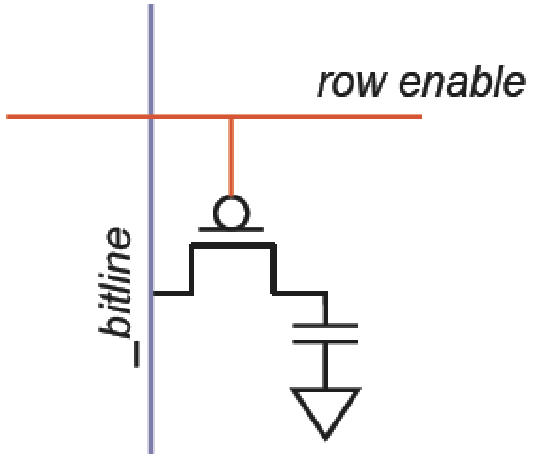
SRAM vs. DRAM

Static Random Access Mem.



- 6T vs. 1T1C
  - Large (~6-10x)
- Bitlines driven by transistors
  - Fast (~10x)

Dynamic Random Access Mem.



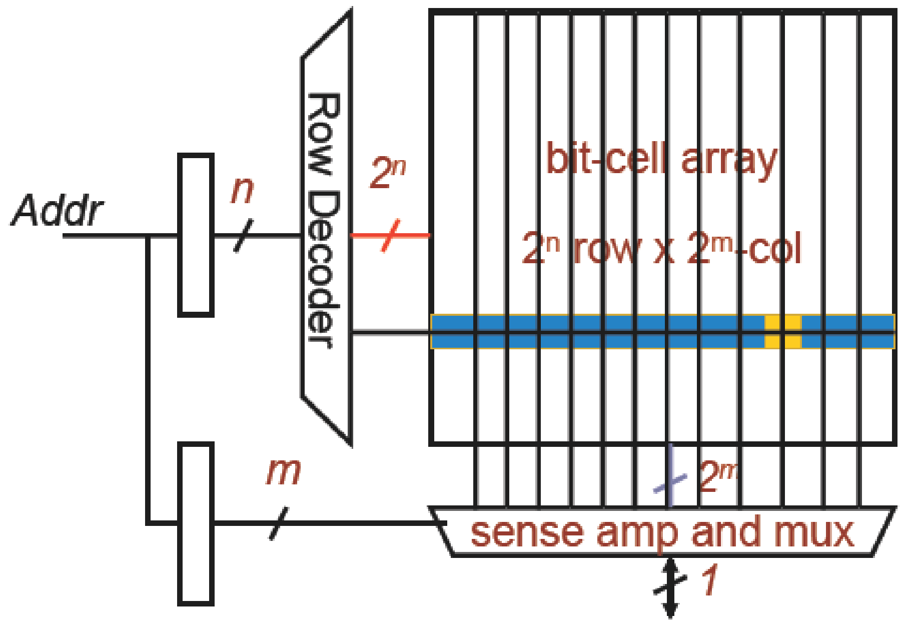
- Bits stored as charges on node capacitance (non-restorative)
  - Bit cell loses charge when read (destructive read)
  - Bit cell loses charge over time
- Must periodically refresh
  - Once every 10s of ms

SRAM vs. DRAM

- SRAM is preferable for register files and L1/L2 caches
  - Fast access
  - No refreshes to worry about
  - Simpler manufacturing (compatible with logic process)
  - Lower density (6 transistors per cell)
  - Higher cost

- DRAM is preferable for stand-alone memory chips
  - Much higher capacity
  - Higher density
  - Lower cost

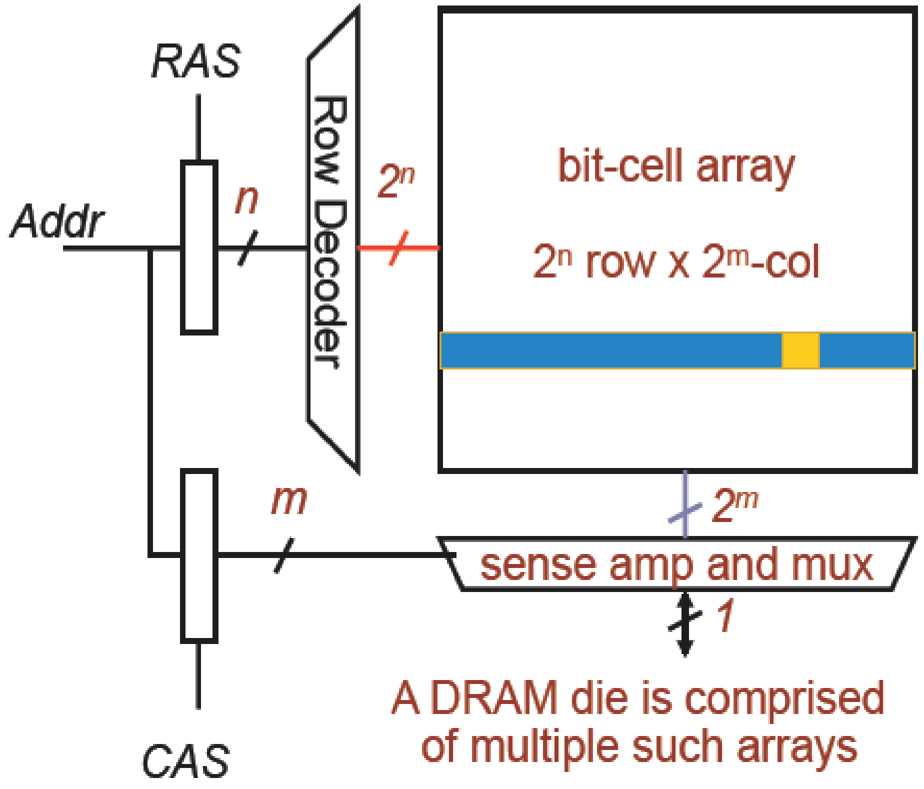
Background: Memory Bank Organization



A DRAM die is comprised of multiple such arrays

- Read access sequence
  - Decode row address & drive word-lines
  - Selected bits drive bit-lines
    - Entire row read
  - Amplify row data
  - Decode column address & select subset of row
  - Send to output
  - Precharge bit-lines for next access

DRAM: Memory-Access Protocol

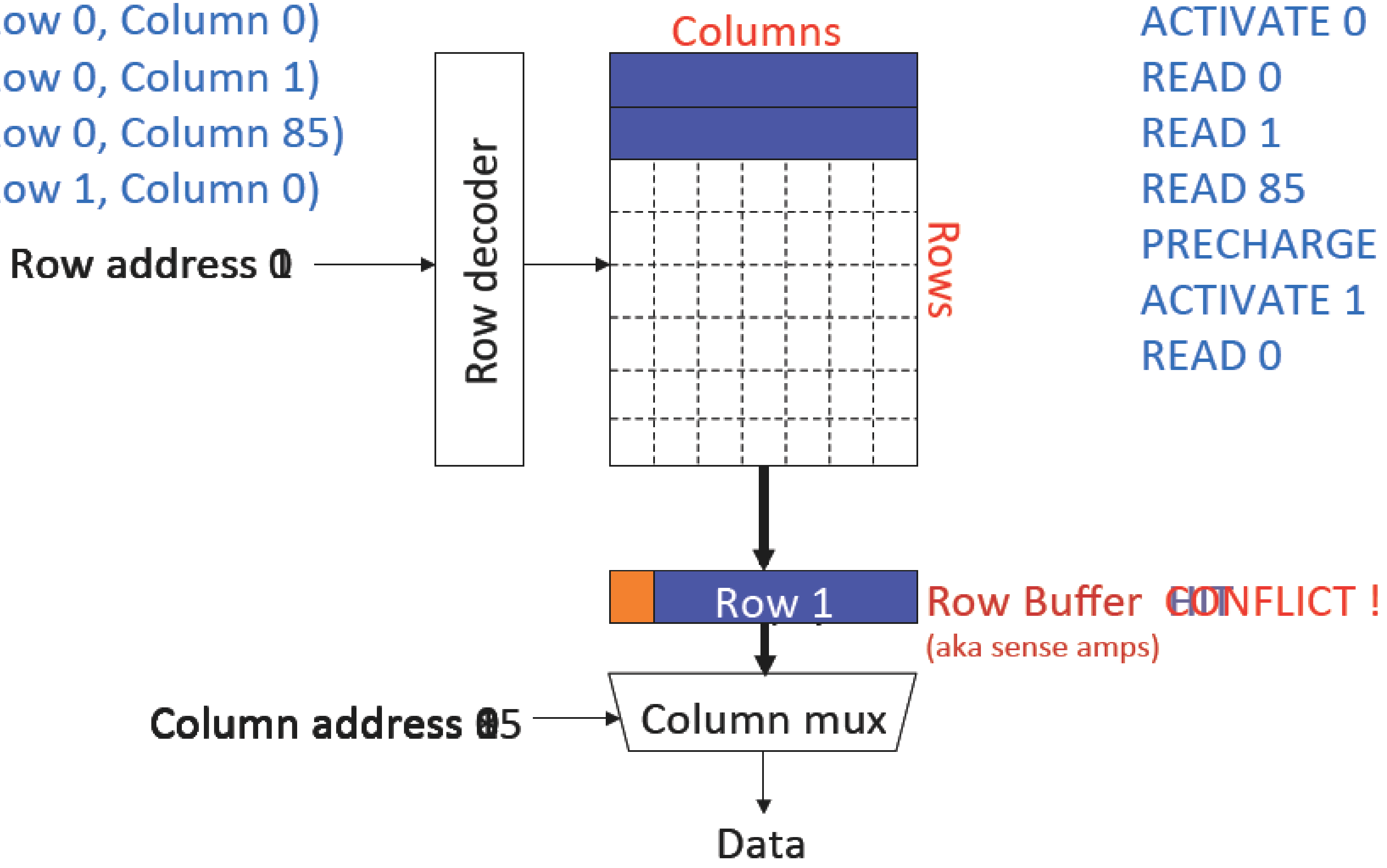


A DRAM die is comprised of multiple such arrays

- 5 basic commands
  - ACTIVATE (open a row)
  - READ (read a column)
  - WRITE
  - PRECHARGE (close row)
  - REFRESH
- To reduce pin count, row and column share same address pins
  - RAS = Row Address Strobe
  - CAS = Column Address Strobe

DRAM: Basic Operation

Addresses	Commands
(Row 0, Column 0)	ACTIVATE 0
(Row 0, Column 1)	READ 0
(Row 0, Column 85)	READ 1
(Row 1, Column 0)	READ 85
	PRECHARGE
	ACTIVATE 1
	READ 0



DRAM: Basic Operation

- Access to an “open row”
  - No need for ACTIVATE command
  - READ/WRITE to access row buffer

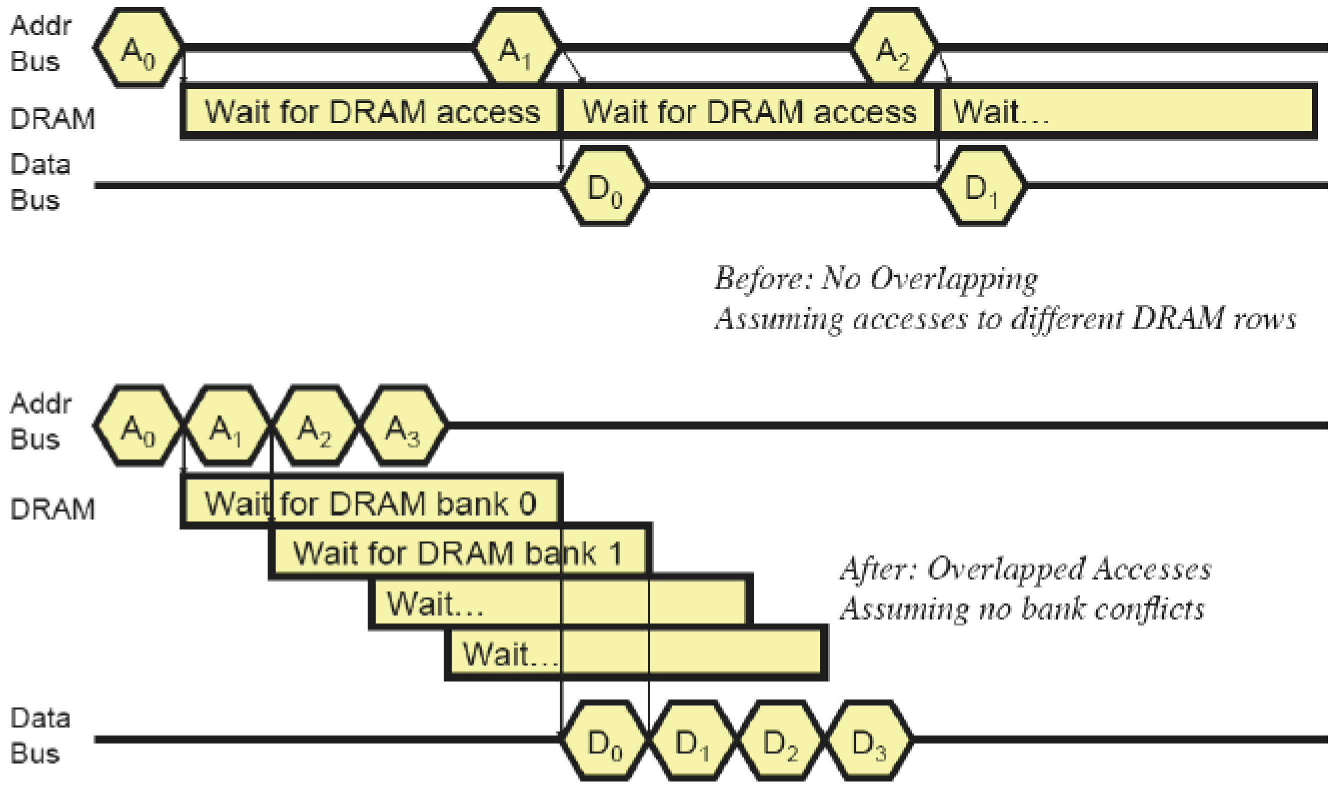
- Access to a “closed row”
  - If another row already active, must first issue PRECHARGE
  - ACTIVATE to open new row
  - READ/WRITE to access row buffer
  - Optional: PRECHARGE after READ/WRITEs finished

DRAM: Banks

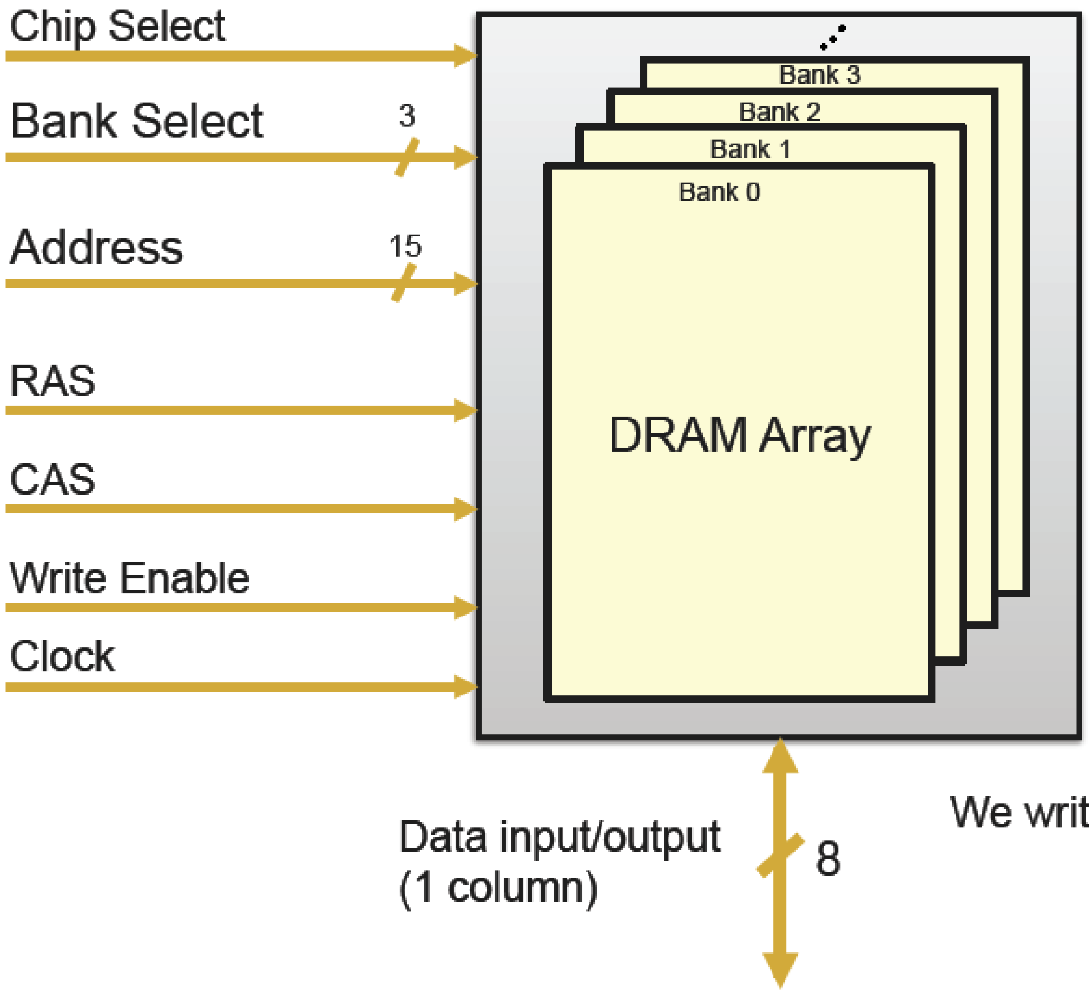
- Modern DRAM chips consist of multiple **banks**
  - Address = (Bank x, Row y, Column z)
- Banks operate independently, but share command/address/data pins
  - Each can have a different row active
  - Can overlap ACTIVATE and PRECHARGE latencies! (i.e. READ to bank 0 while ACTIVATING bank 1)

DRAM: Banks

- Enable concurrent DRAM accesses (overlapping)

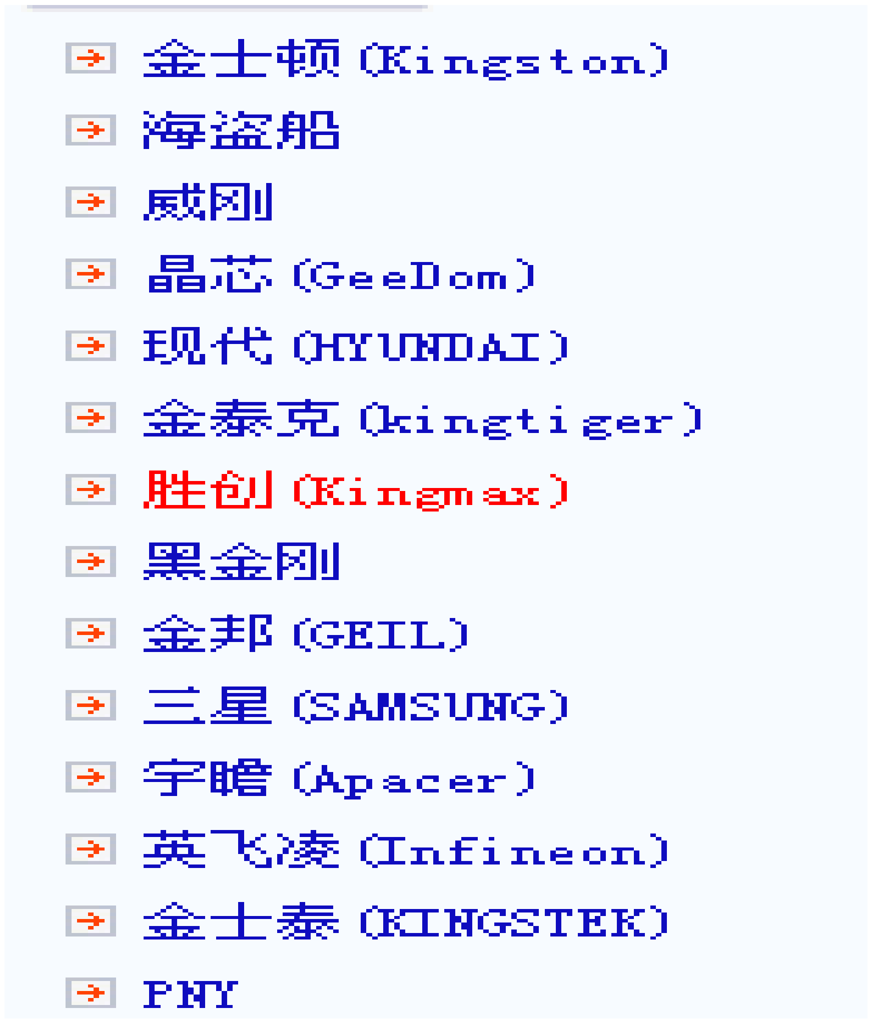


DRAM Chip: High-level View



We write "x8" to mean an 8 data bits, x16 = 16 data bits, etc.

内存厂商&内存颗粒制造商



- 金士顿(Kingston)
- 海盗船
- 威刚
- 晶芯(GeeDom)
- 现代(HYUNDAI)
- 金泰克(kingtiger)
- 胜创(Kingmax)
- 黑金刚
- 金邦(GEIL)
- 三星(SAMSUNG)
- 宇瞻(Apacer)
- 英飞凌(Infineon)
- 金士泰(KINGSTEK)
- PNY

内存颗粒制造:

三星(SAMSUNG)

现代(Hynix)

英飞凌(Infineon)

美光(Micron)

勤茂(TwinMOS)

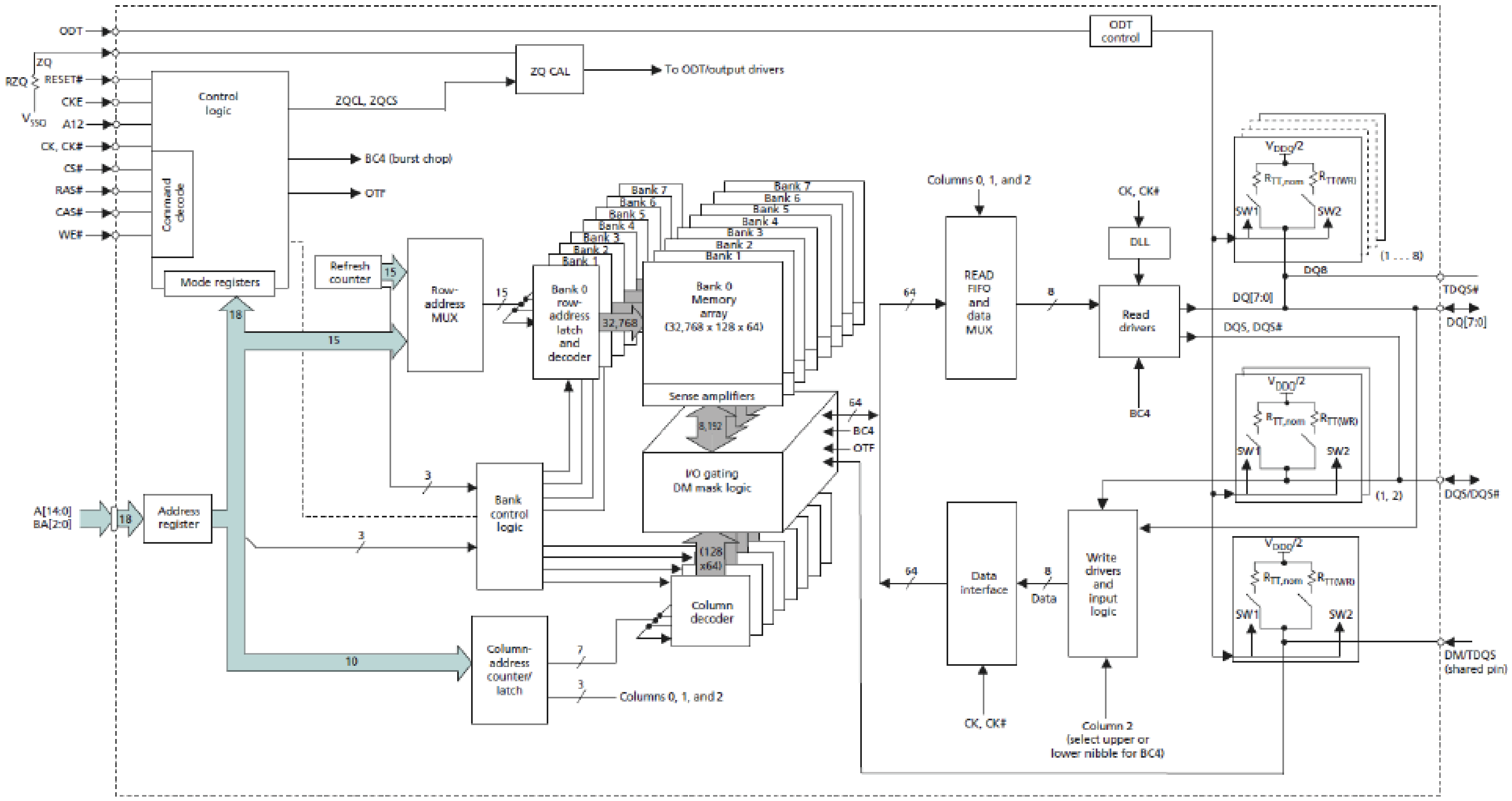
南亚(NANYA)

华邦(Winbond)

茂矽(MOSEL)

ELPIDA等半导体厂商

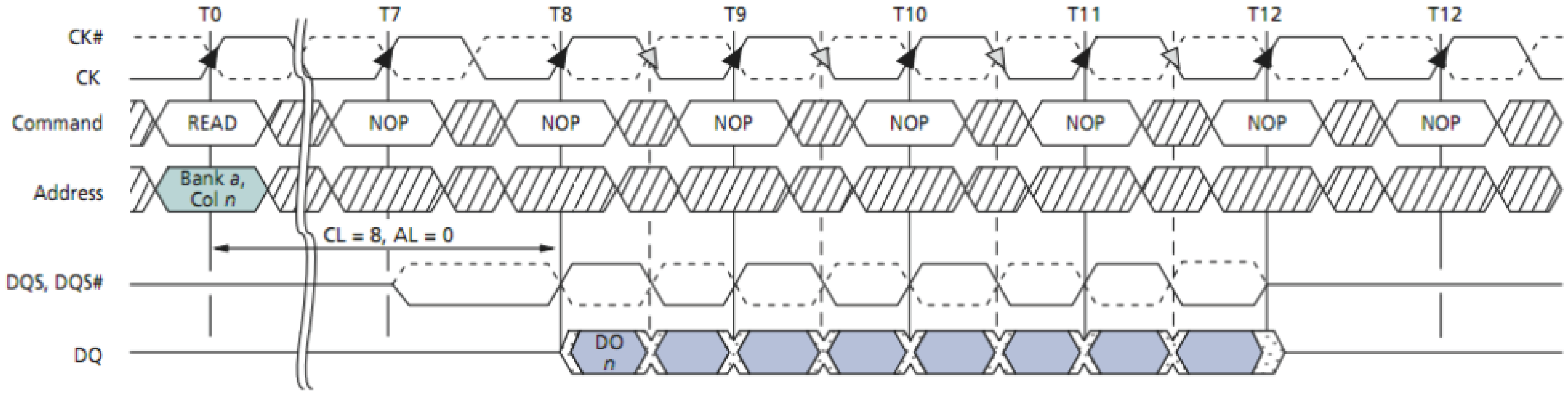
2Gb x8 DDR3 Chip [Micron]



Observe: bank organization

DRAM: Burst

- Each READ/WRITE command transfers multiple columns (8 in DDR3)
- DRAM channel clocked faster than DRAM core



- Critical word first?

Industry standard: DDR3 SDRAM

- Introduced in 2007
- SDRAM = Synchronous DRAM = **Clocked**
- DDR = Double Data Rate
  - Data transferred on both clock edges
  - 400 MHz = 800 MT/s
- x4, x8, x16 datapath widths
- Minimum burst length of 8
- 8 banks
- 1Gb, 2Gb, 4Gb, 8Gb capacity common
- Relative to SDR/DDR/DDR2: + bandwidth, ~ latency

DDR4

- 2011年1月4日，三星生产出第一条DDR4内存。
- DDR4相比DDR3最大的区别有三点：16bit预取机制（DDR3为8bit），同样内核频率下理论速度是DDR3的两倍；
- 更可靠的传输规范，数据可靠性提升；
- 工作电压降为1.2V，更节能。//DDR3是1.5V

DDR5

- 2020年1月7日，美光出样DDR5内存，2021年正式上市，Intel和AMD已有服务器级应用。
- 比DDR4性能提升85%，性能更强，功耗更低
- 标准频率4800MHz，最高6400MHz
- 电压从1.2V降低到1.1V，总线效率提高。

DRAM: Timing Constraints

- Memory controller must respect physical device characteristics
  - **tRCD** = Row to Column command delay
    - How long it takes row to get to sense amps
  - **tCAS** = Time between column command and data out
  - **tCCD** = Time between column commands
    - Rate that you can pipeline column commands
  - **tRP** = Time to precharge DRAM array
  - **tRAS** = Time between RAS and data restoration in DRAM array (minimum time a row must be open)
  - **tRC** = tRAS + tRP = Row "cycle" time
    - Minimum time between accesses to different rows

DRAM: Timing Constraints

- There are dozens of these…
  - tWTR = Write to read delay
  - tWR = Time from end of last write to PRECHARGE
  - tFAW = Four ACTIVATE window (limits current surge)
- Makes performance analysis, memory controller design difficult
- Datasheets for DRAM devices freely available
  - http://download.micron.com/pdf/datasheets/dram/ddr3/2Gb_DDR3_SDRAM.pdf

tFAW

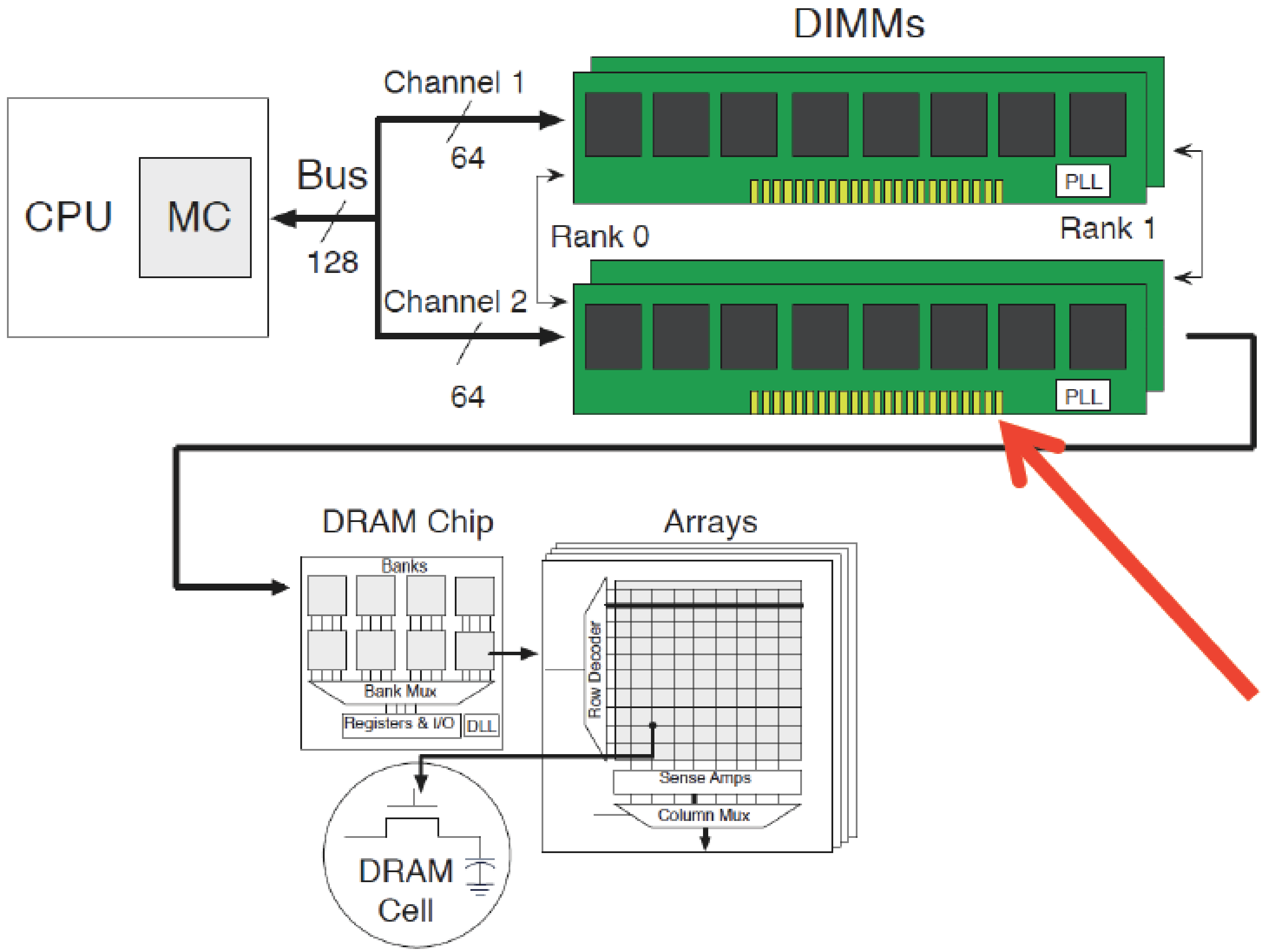
- DRAM Four Active Window的缩写，DDR内存的一个时序参数，定义了同一Rank中允许同时发送大于四个行激活命令的间隔时间，因此最小值应该不小于tRRD(RAS to RAS Delay)的四倍。

- tFAW会限制服务器的数据吞吐量，严重的时候可以让带宽打个15-35％的折扣。DDR内存每进步一代，访问间隔就会翻番，导致tRDD、tFAW这样的时序参数越发限制数据吞吐量，影响高性能应用的发挥，因为在任何tFAW期间都不能同时发送超过四个激活命令。

Latency Components:Basic DRAM Operation

- CPU → memory controller transfer time
- Controller latency
  - Queuing & scheduling delay at the controller
  - Access converted to basic commands
- DRAM bank latency
  - tCAS if row is "open" OR
  - tRCD + tCAS if array precharged OR
  - tRP + tRCD + tCAS (worst case: tRC + tRCD + tCAS)
- DRAM data transfer time
  - BurstLen / (MT/s)
- Memory controller → CPU transfer time

Main Memory Overview

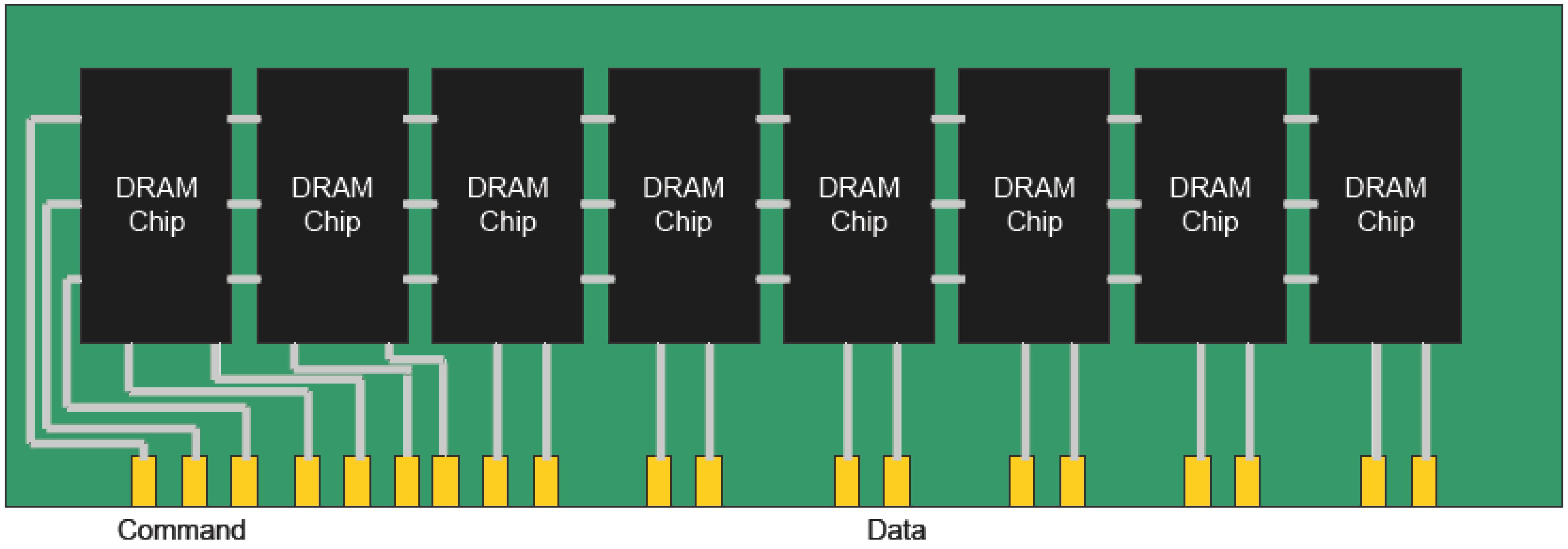


DRAM Modules

- DRAM chips have narrow interface (typically x4, x8, x16)
- Multiple chips are put together to form a wide interface
  - DIMM: Dual Inline Memory Module
  - To get a 64-bit DIMM with x8 DRAM chips, we need to access 8 chips
  - Share command/address lines, but not data

- Advantages
  - Acts like a high-capacity DRAM chip with a wide interface
    - 8x capacity, 8x bandwidth, same latency
- Disadvantages
  - Granularity: Accesses cannot be smaller than the interface width
  - 8x power

A 64-bit Wide DIMM (physical view)

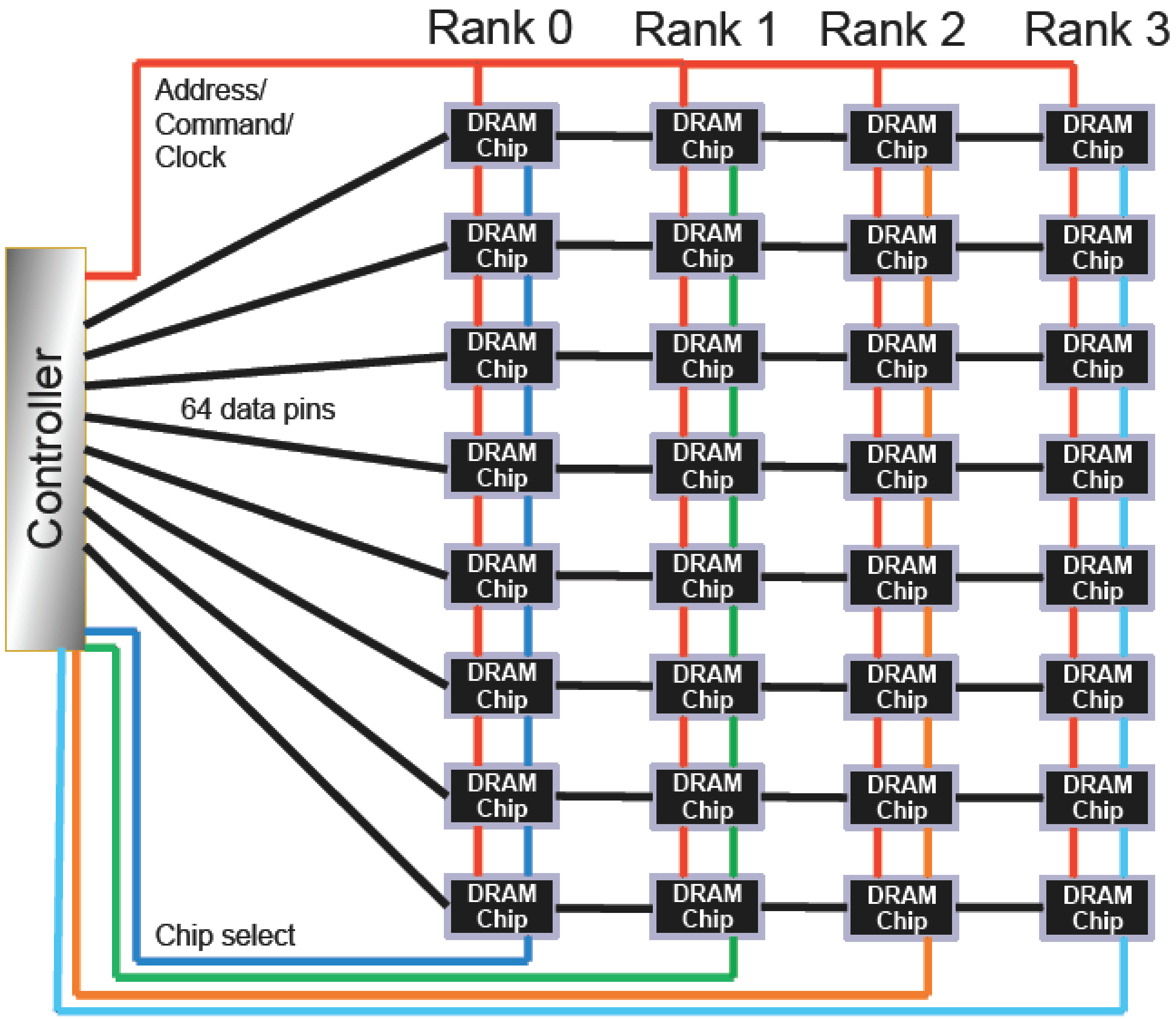


DIMM Capacity

- 64-bit interface
- 2Gb x8 chips
- 64/8 = 8 chips
- 2Gb x 8 = 2GB

- What if we want more capacity on a channel?
  - Option 1: Use narrower chips (i.e. x4)
  - Option 2: Ranks

Ranks

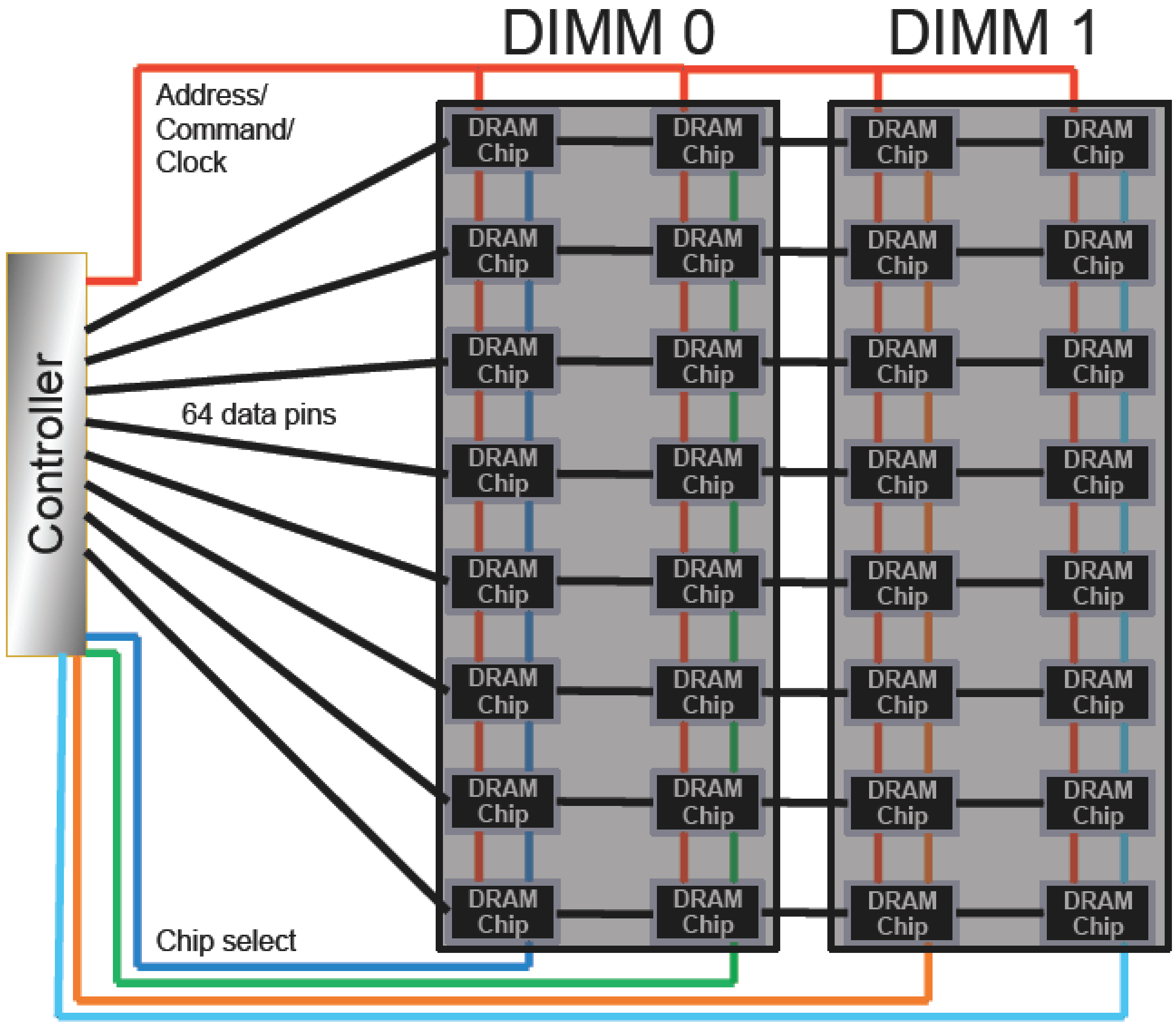

Rank 0
Rank 1
Rank 2
Rank 3
Address/
Command/
Clock
Controller
64 data pins
Chip select
DRAM Chip

Ranks

- A DIMM may include multiple Ranks
  - A 64-bit DIMM with 16 chips with x8 interfaces has 2 ranks

- Each 64-bit group of chips is called a rank
  - All chips in a rank respond to a single command
  - Different ranks share command/address/data lines
    - Select between ranks with "Chip Select" signal
  - Ranks provide more "banks" across multiple chips
    (but don't confuse rank and bank!)

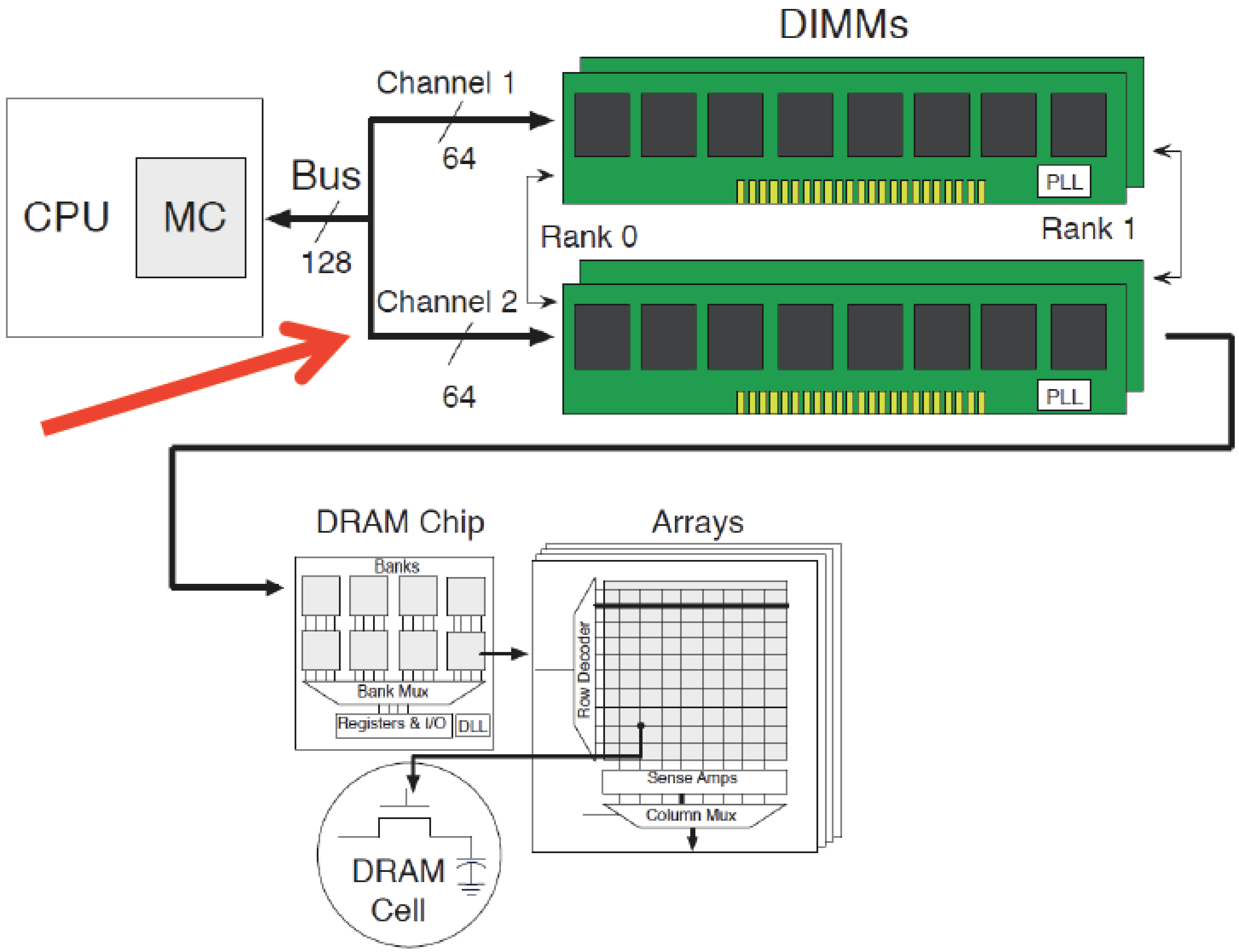
Multiple DIMMs on a Channel



DRAM Capacity

- What if we want even more capacity?
  - Can only put limited number of ranks on channel (signal and driver limitations)
- x2 chips? x1 chips?
  - What happens to power consumption?
- More channels?

Main Memory Overview

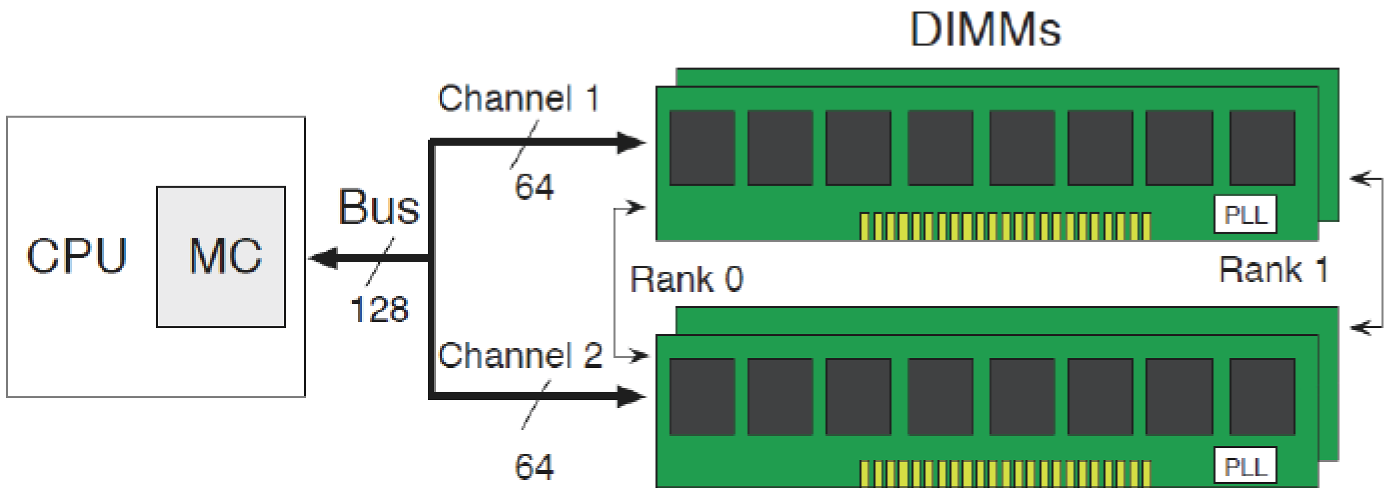


DRAM Channels

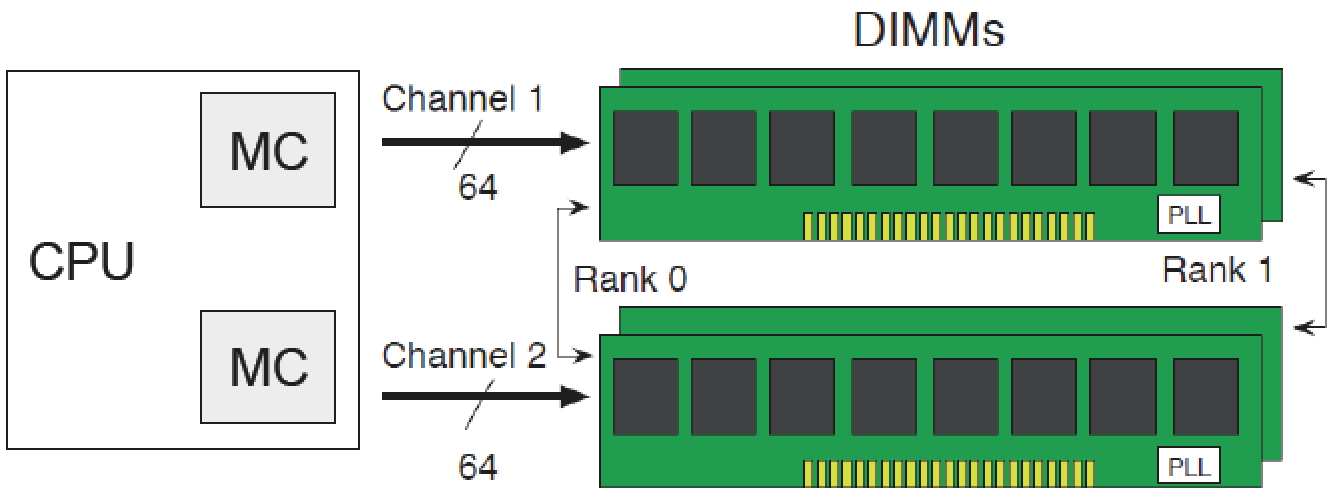
- Channel: a set of DIMMs in series
  - All DIMMs get the same command, one of the ranks replies
- Multiple-channel options
  - Multiple lock-step channels
    - Single “channel” with wider interface (faster cache line refill!)
    - Sometimes called “Gang Mode”
    - Only works if DIMMs are identical (organization, timing)
  - Multiple independent channels
    - Requires multiple controllers
- Tradeoffs in having multiple channels
  - Cost: pins, wires, controllers
  - Benefit: higher bandwidth, capacity

DRAM Channel Options

Lock-step



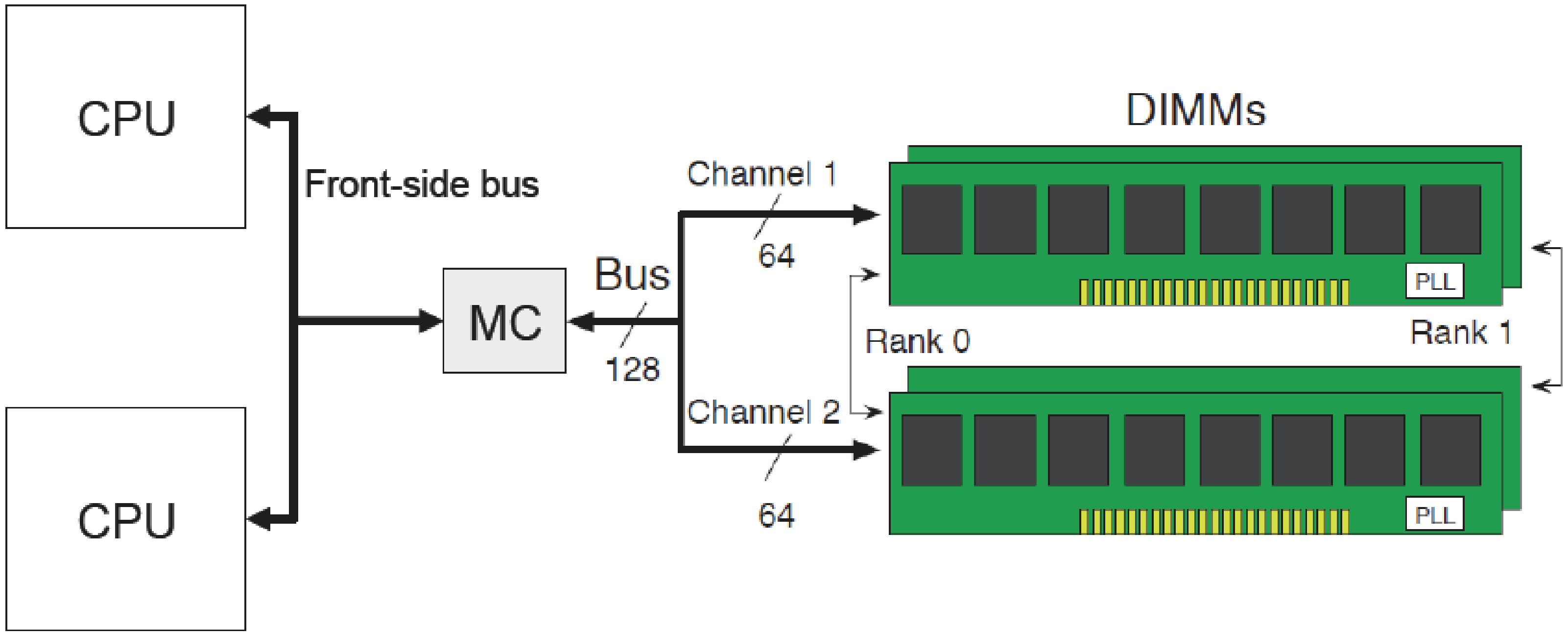
Independent



Lock-step DRAM Channels

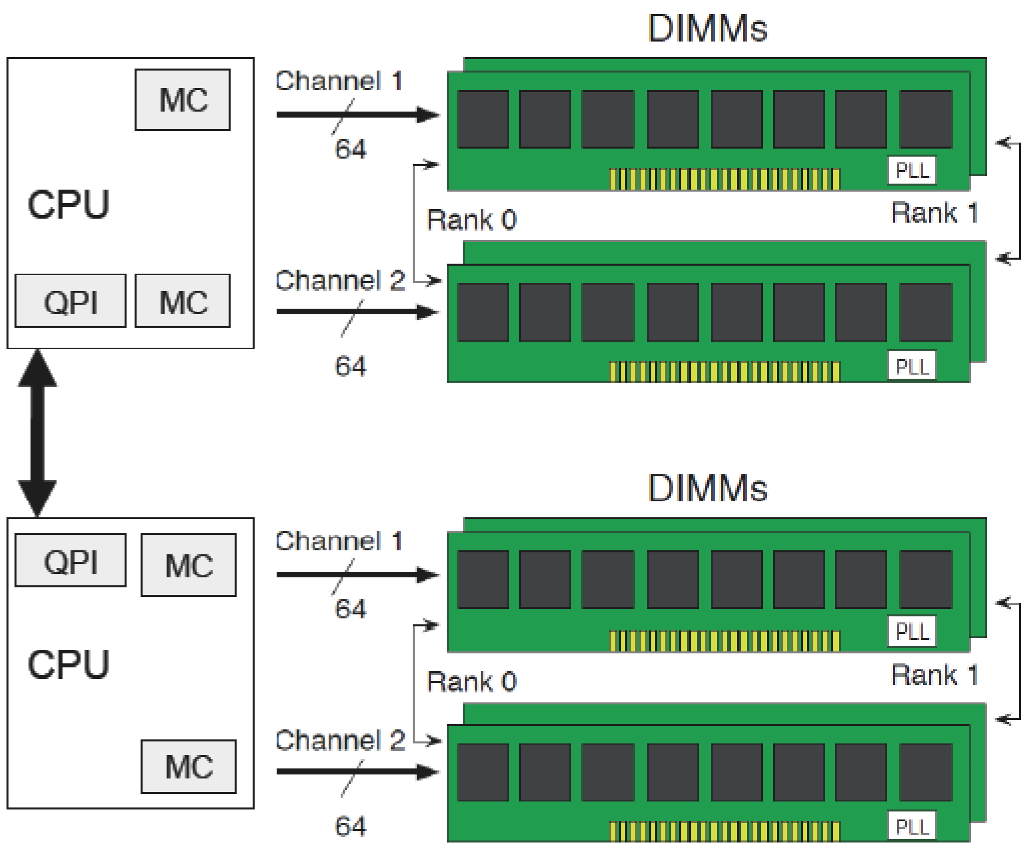
- Assume a lock-step dual channel bus
  - Two 64-bit channels
  - Minimum burst-length of 8 (i.e. DDR3)
- What's the minimum transfer size?
  - Implication?

Multi-CPU (old school)



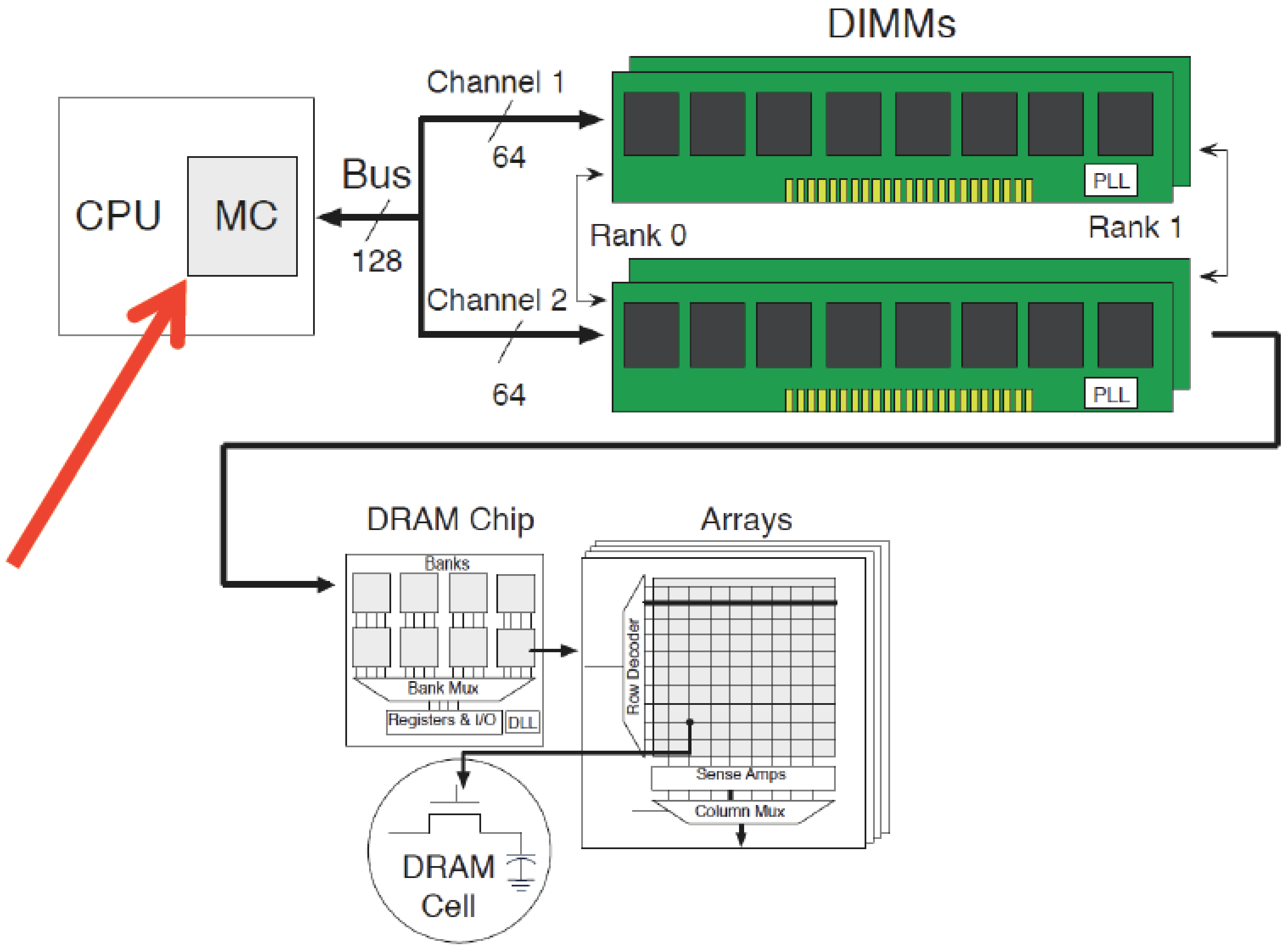
- External MC adds latency
- CPUs compete for memory bandwidth

NUMA Topology (modern)



- Capacity and bandwidth grows w/ # of CPUs
- NUMA: “Non-uniform Memory Access”
  - What are the performance implications?

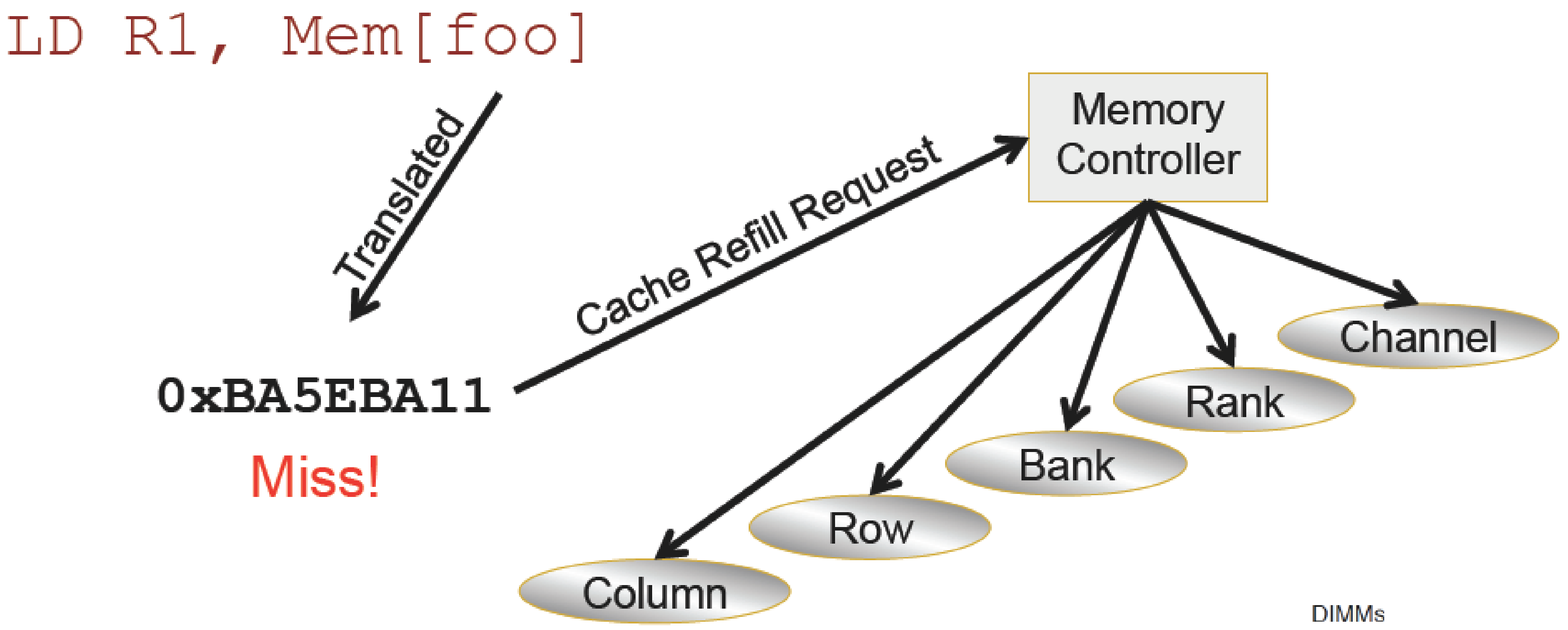
Main Memory Overview

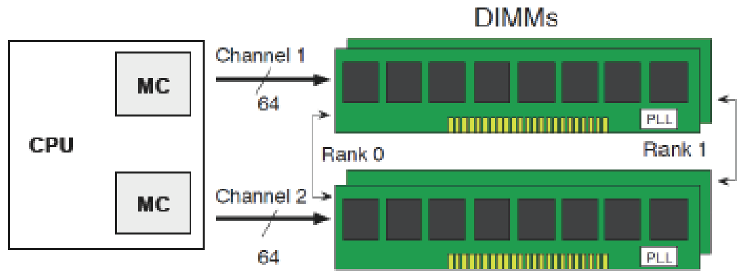


DRAM Controller Functionality

- Translate memory requests into DRAM command sequences
  - Map "Physical Address" to DRAM Address
  - Obey timing constraints of DRAM, arbitrate resource conflicts (i.e. bank, channel)
- Buffer and schedule requests to improve performance
  - Row-buffer management and re-ordering
- Ensure correct operation of DRAM (refresh)
- Manage power consumption and thermals in DRAM
  - Turn on/off DRAM chips, manage power modes

DRAM Addressing





Address Mapping (Single Channel)

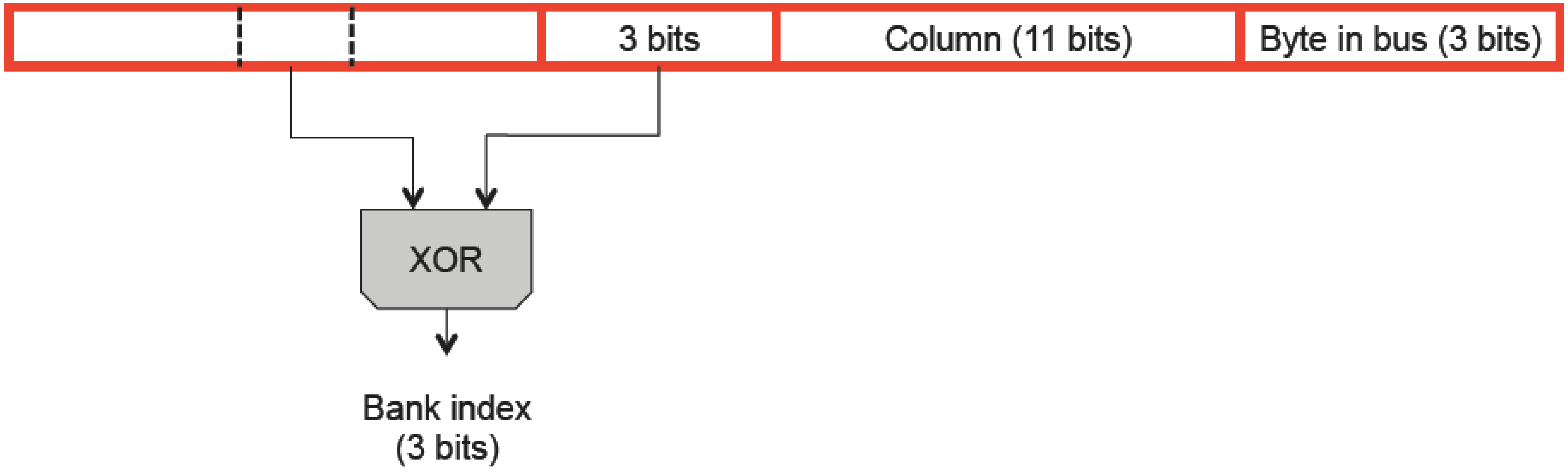
- Assume a single-channel system with 8-byte memory bus
  - 2GB DIMM, 1 rank, 8 banks
  - 16K rows & 2K columns per bank
  - 64 byte cache block
- Row interleaving
  - Consecutive rows of memory in consecutive banks

Address (i.e. `0xBA5EBA11`)

Row (14 bits)	Bank (3 bits)	Column (11 bits)	Byte in bus (3 bits)

Bank Mapping Randomization

- DRAM controller can randomize the address mapping to banks so that bank conflicts are less likely



Address Mapping (Multiple Channels)

C	Row (14 bits)	Bank (3 bits)	Column (11 bits)	Byte in bus (3 bits)

Row (14 bits)	C	Bank (3 bits)	Column (11 bits)	Byte in bus (3 bits)

Row (14 bits)	Bank (3 bits)	C	Column (11 bits)	Byte in bus (3 bits)

Row (14 bits)	Bank (3 bits)	Column (11 bits)	C	Byte in bus (3 bits)

Lock-step

- Cache block interleaving

Row (14 bits)	Bank (3 bits)	High Col.	C	Low Col.	Byte in bus (3 bits)
		8 bits		3 bits	

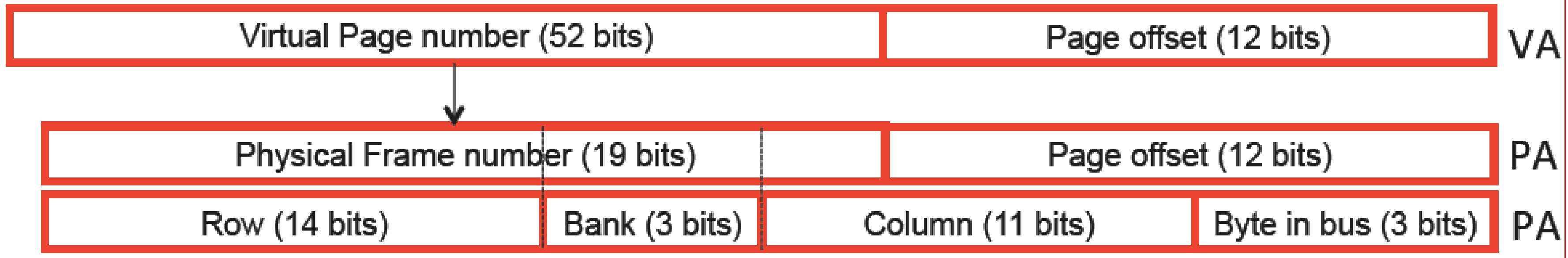
  - What happens with NUMA topology?

- Don’t forget ranks ☺

Row (14 bits)	Rank (2 bits)	Bank (3 bits)	High Col.	C	Low Col.	Byte in bus (3 bits)
			8 bits		3 bits	

Interaction with Virtual → Physical Mapping

- OS influences where an address maps to in DRAM



- OS can control which bank a virtual page is mapped to
  - Randomize Page→<Bank,Channel> for load-balancing
  - Optimize page placement for NUMA locality

- App does't know which bank/channel it is accessing

Reminder: Row Access Latency

- Access to an "open row"
  - No need for ACTIVATE command
  - READ/WRITE to access row buffer
- Access to a "closed row"
  - If another row already active, must first issue PRECHARGE
  - ACTIVATE to open new row
  - READ/WRITE to access row buffer
  - Optional: PRECHARGE after READ/WRITEs finished

Row Buffer Management Policies

- Open row policy
  - Keep the row open after an access (i.e. don't precharge)
  - Pro: Next access might need the same row → row hit! Success.
  - Con: Next access might need a different row → row conflict, extra latency

- Closed row policy
  - Close the row immediately after an access
    (if no other requests are already waiting for the same row)
  - Pro: Next access might need a different row → avoided the row conflict!
  - Con: Next access might need the same row → extra latency, wasted energy

- Adaptive policies
  - Any row-buffer policy is ***speculative***
  - Predict whether or not the next access to the bank will be to the same row

Open vs. Closed Row Policies

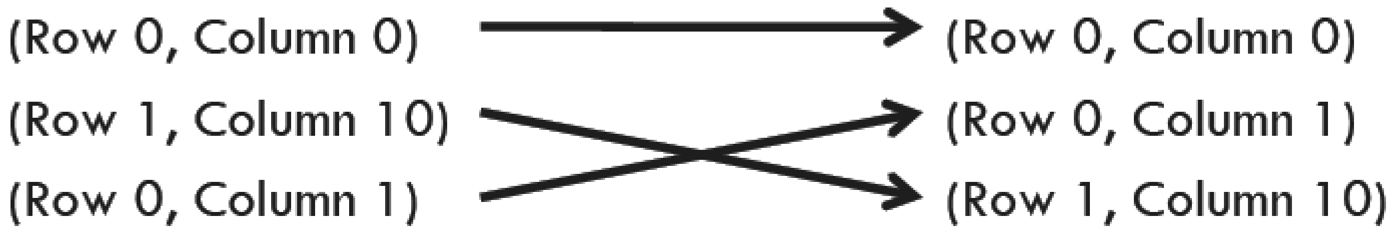
Policy	First access	Next access	Commands needed for next access
Open row	Row 0	Row 0 (row hit)	Read
Open row	Row 0	Row 1 (row conflict)	Precharge + Activate Row 1 + Read
Closed row	Row 0	Row 0 – access in request buffer (row hit)	Read
Closed row	Row 0	Row 0 – access not in request buffer (row closed)	Activate Row 0 + Read + Precharge
Closed row	Row 0	Row 1 (row closed)	Activate Row 1 + Read + Precharge

DRAM Scheduling Policies (I)

- FCFS (first come first served)
  - Oldest request first

- FR-FCFS (first ready, first come first served)
  - 1. Row-hit first
  - 2. Oldest first
  - Goal: Maximize row buffer hit rate $\rightarrow$ maximize DRAM throughput

DRAM Scheduling Example



FCFS	FR-FCFS
ACTIVATE 0	ACTIVATE 0
READ 0	READ 0
PRECHARGE	READ 1
ACTIVATE 1	PRECHARGE
READ 10	ACTIVATE 1
PRECHARGE	READ 10
ACTIVATE 0	
READ 1	

DRAM Scheduling Policies (II)

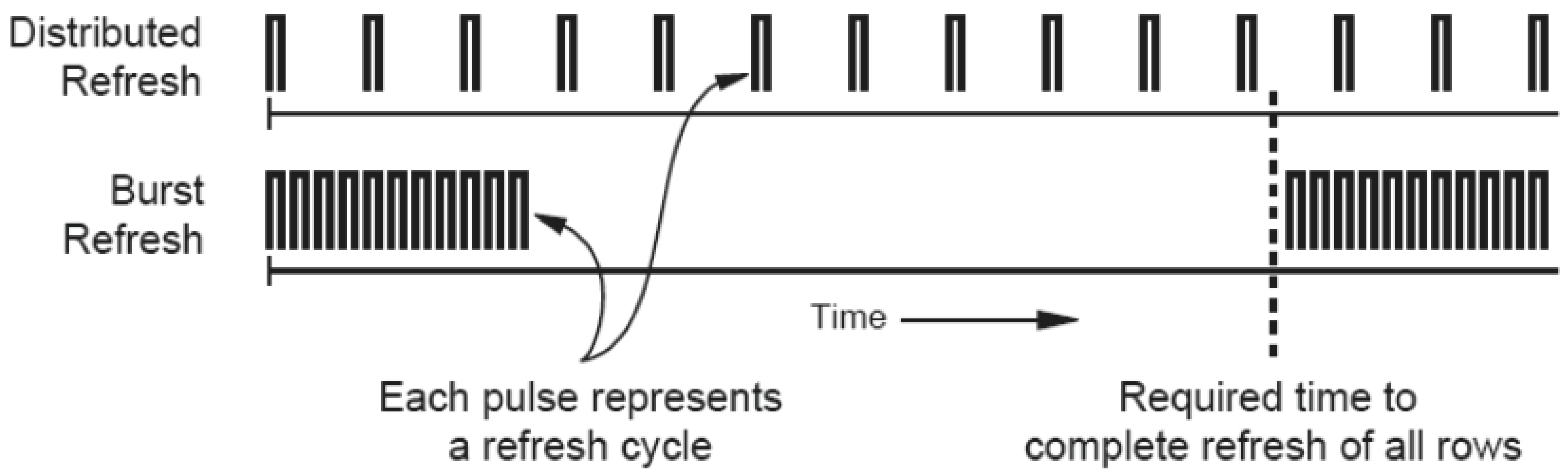
- A scheduling policy is a prioritization order

- Prioritization can be based on
  - Request age (FCFS)
  - Row buffer hit/miss status (FR-FCFS)
  - Request type (prefetch, read, write)
  - Requestor type (load miss or store miss)
  - Request criticality
    - Oldest miss in the core?
    - How many instructions in core are dependent on it?

DRAM Refresh (I)

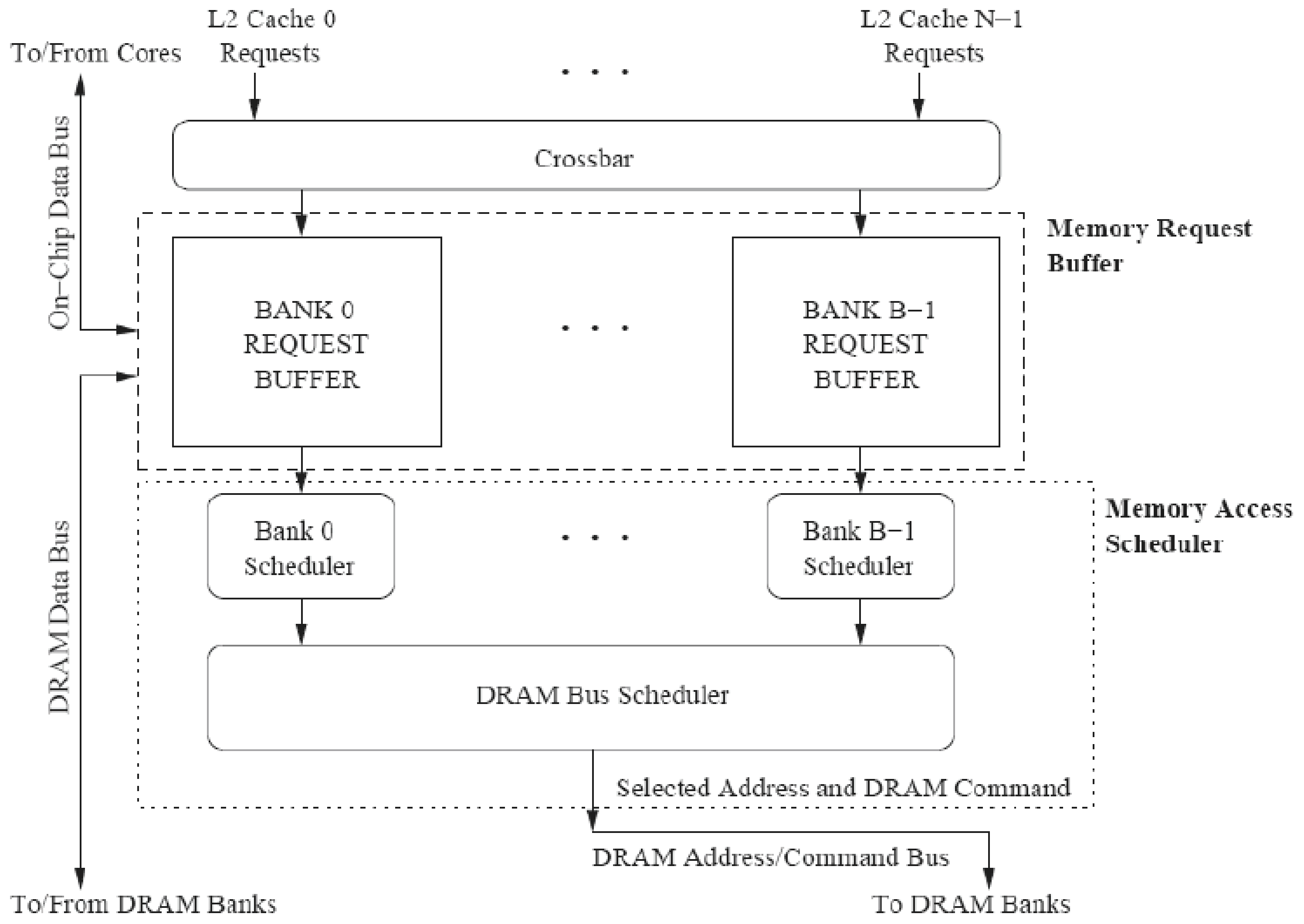
- DRAM capacitor charge leaks over time
- The memory controller needs to read each row periodically to restore the charge
  - Refresh = Activate + Precharge each row every N ms
  - Typical N = 64 ms
  - REFRESH command: DRAM chip picks a row to refresh
- Implications on performance?
  - DRAM bank unavailable while refreshed

DRAM Refresh (II)


Distributed Refresh
Burst Refresh
Time
Each pulse represents a refresh cycle
Required time to complete refresh of all rows

- Distributed refresh eliminates long pause times
- How else we can reduce the effect of refresh on performance?
  - Can we reduce the number of refreshes?

A Modern DRAM Controller



DRAM Controllers are Complex

- Need to obey DRAM timing constraints for correctness
  - There are many (50+) timing constraints in DRAM
- Need to keep track of many resources to prevent conflicts
  - Channels, banks, ranks, data bus, address bus, row buffers
- Need to handle DRAM refresh
- Need to optimize for performance (in the presence of constraints)
  - Reordering is not simple
  - Predicting the future?

致谢：

本讲内容参考了M.I.T. Daniel Sanchez教授的课程讲义，特此感谢。