CMPE 110: Computer Architecture

Week 9
Main Memory/Virtual Memory

Jishen Zhao (http://users.soe.ucsc.edu/~jzhao/)

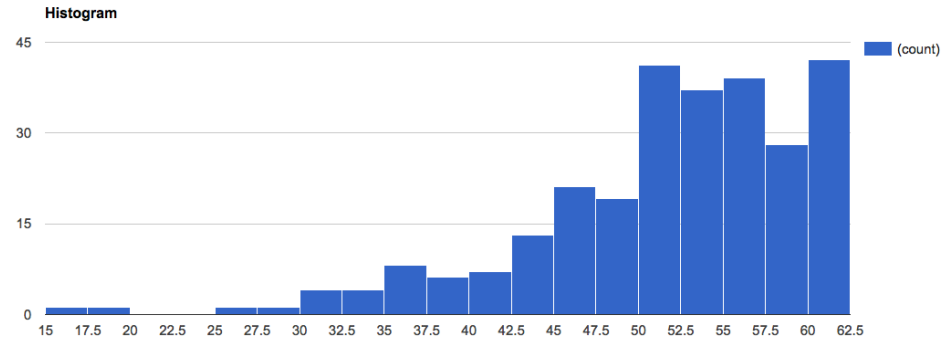
[Adapted in part from Jose Renau, Mary Jane Irwin, Joe Devietti, Onur Mutlu, and others]

Reminder

- Homework 3 is due today 11:59pm
  - Question 3 will be a bonus question
- Homework 4 will be posted on Thursday

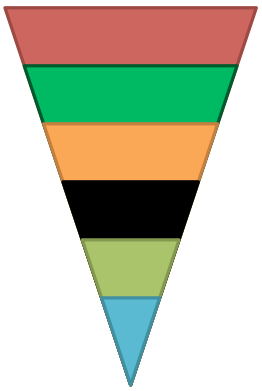
Midterm2 histogram

Max = 60
Min = 17
Mean = 51.4
Median = 53
Standard Deviation = 8



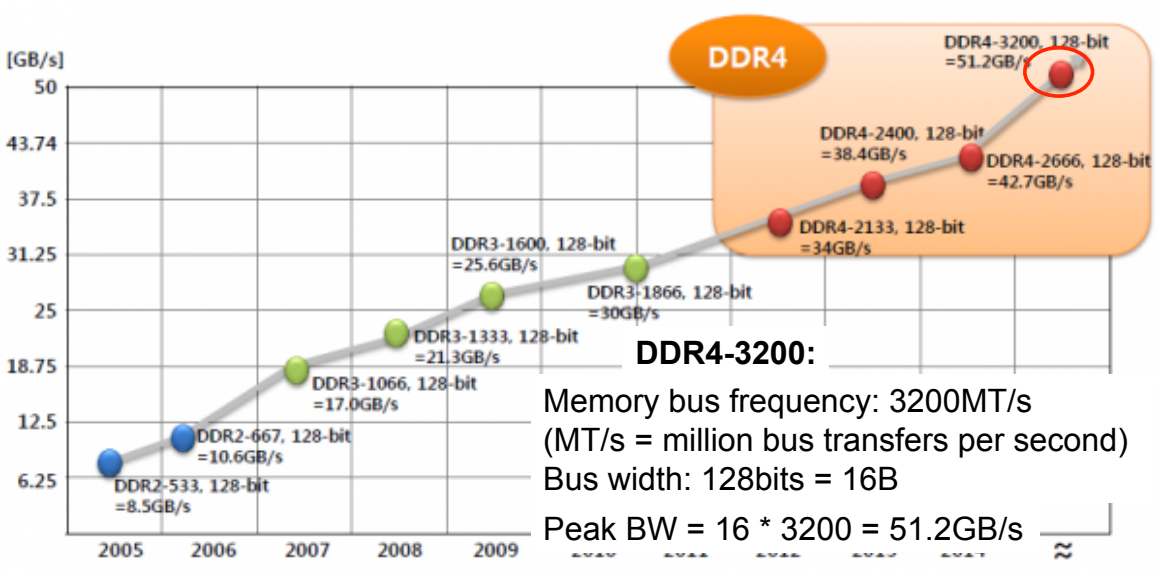
Review: main memory organization

- Channel
- DIMM
- Rank
- Chip
- Bank
- Row/Column



Review: Peak memory bandwidth

Peak memory bandwidth = memory bus width * memory bus frequency

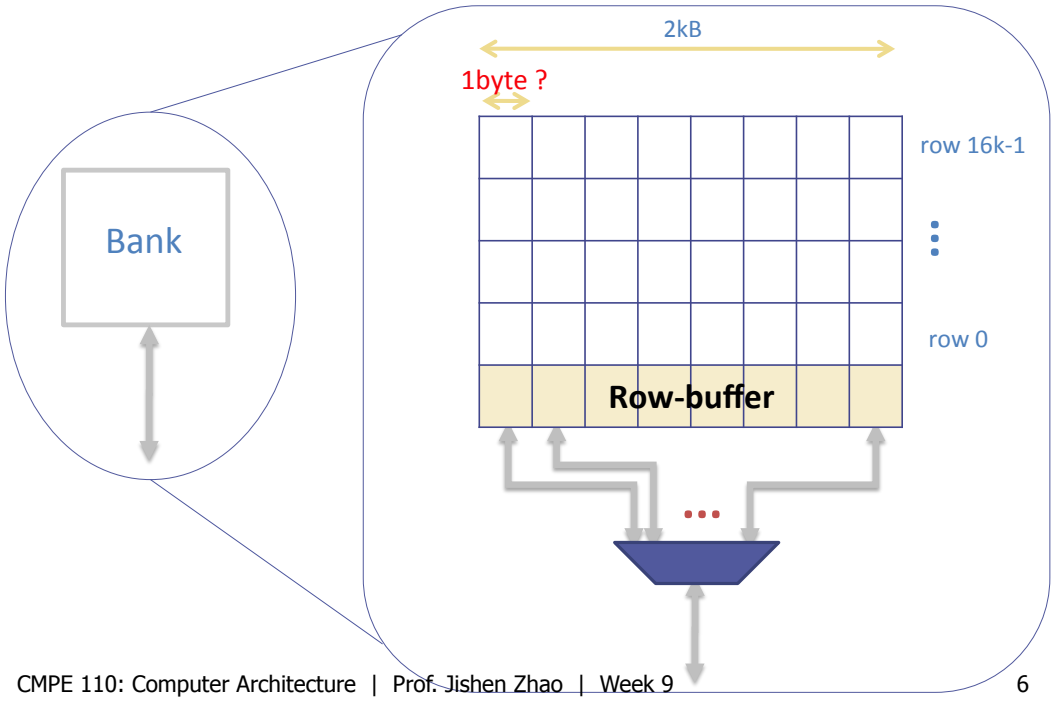


DDR4-3200:

Memory bus frequency: 3200MT/s
(MT/s = million bus transfers per second)
Bus width: 128bits = 16B

Peak BW = 16 * 3200 = 51.2GB/s

Review: row buffer

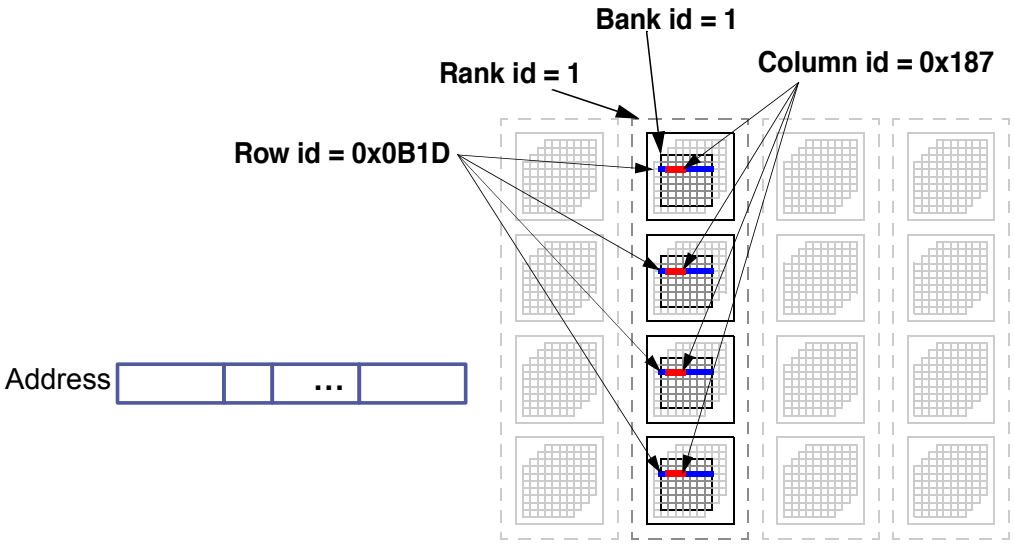


Review: Open row and closed row

- A closed row vs. an open row
  (analogy: write-through vs. write-back cache)
  - A closed row: data are not buffered in the row buffer after each access (similar to "write-through")
  - An open row: data are buffered in the row buffer after access (similar to "write-back")
  - Can only have one open row at a time
- Access to an "open row" – data in the row buffer
  - Read/write command reads/writes column in the row buffer
- Access to a "closed row" – data not in the row buffer
  - Activate command opens row (placed into row buffer)
  - Read/write command reads/writes column in the row buffer
  - Precharge command closes the row and prepares the bank for next access

Today

- Main memory
  - Where is data – how to access main memory using an address
  - Improve parallelism of memory access
  - Address mapping
- Virtual memory

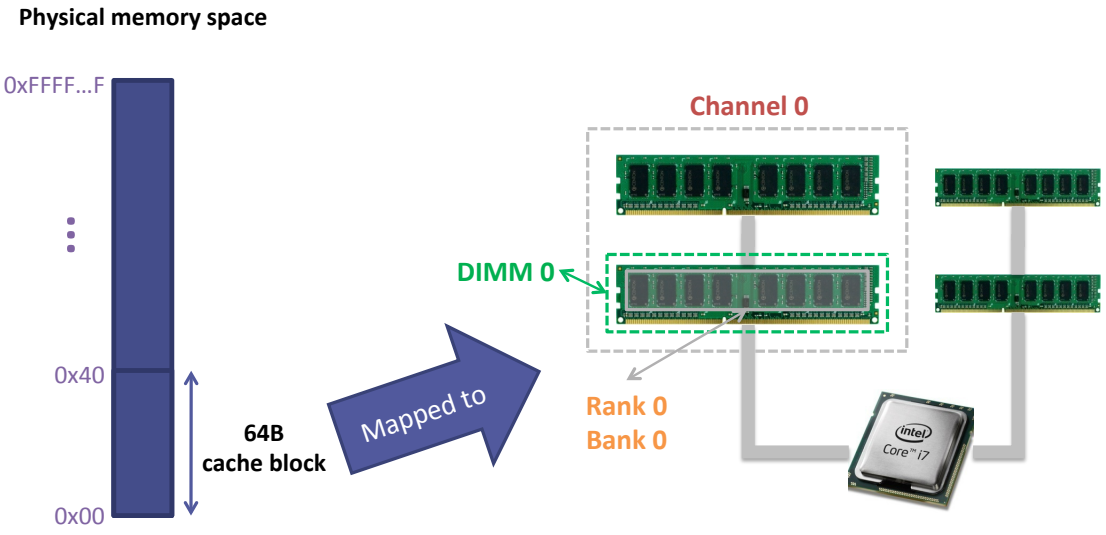


Where is the data

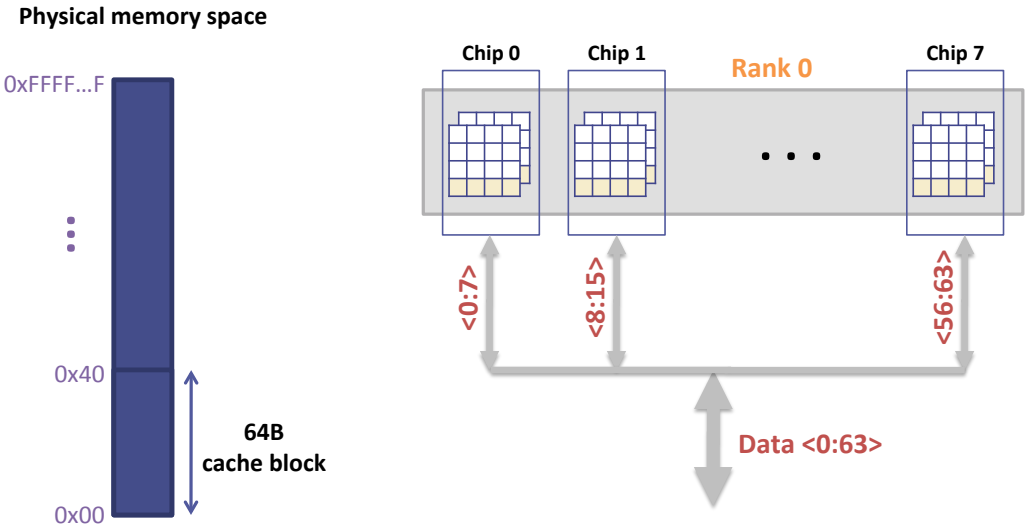
How to access main memory

- Page mode DRAM
  - [Known] A DRAM bank is a 2D array of cells:
    - rows x columns
  - A "DRAM row" is also called a "DRAM page"
- An example
  - Fetch a 64-byte cache block from main memory

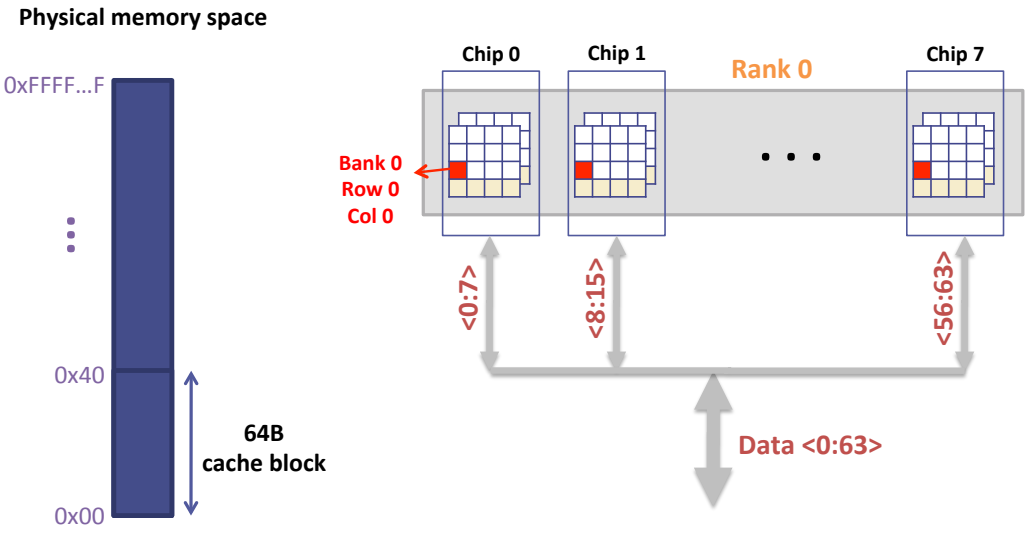
Example: Transferring a cache block



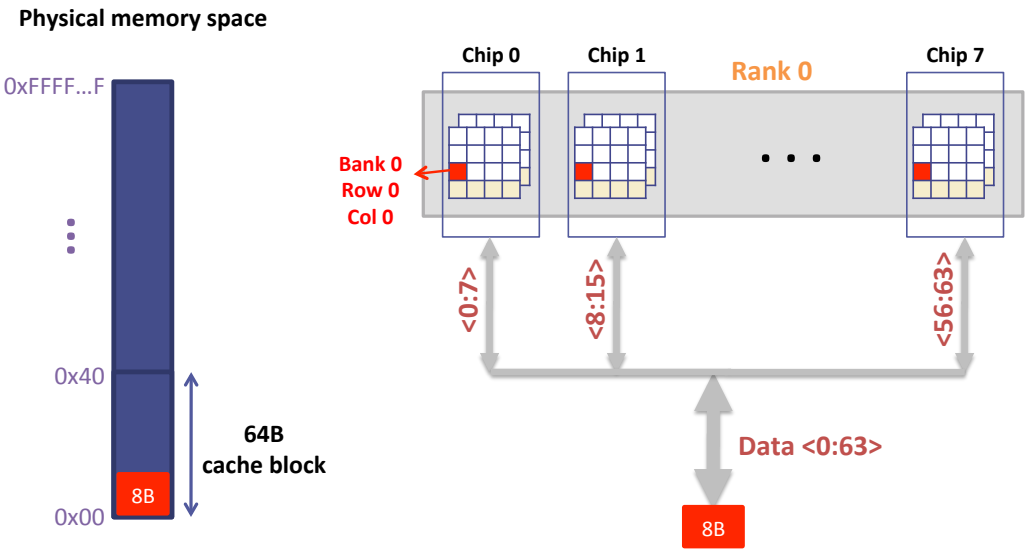
Example: Transferring a cache block



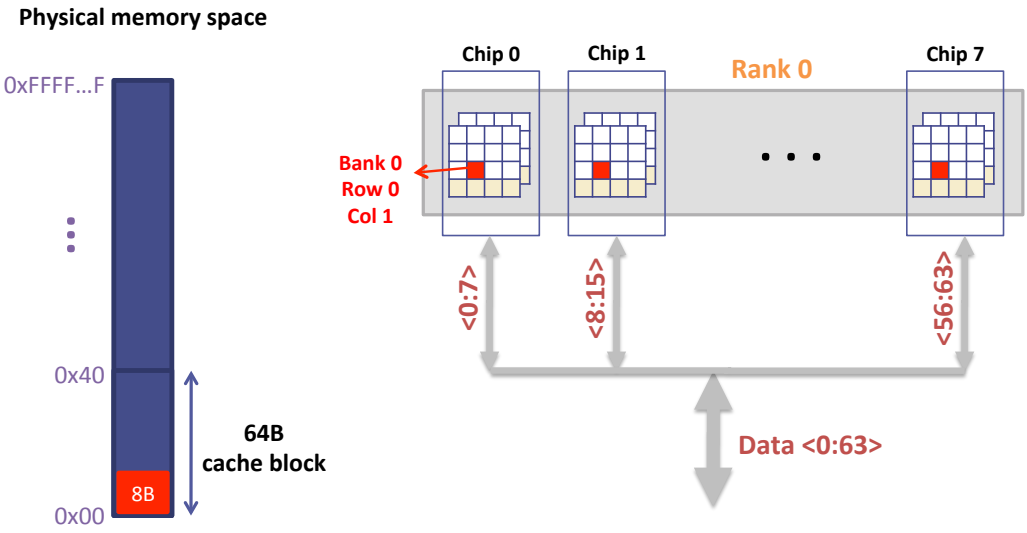
Example: Transferring a cache block



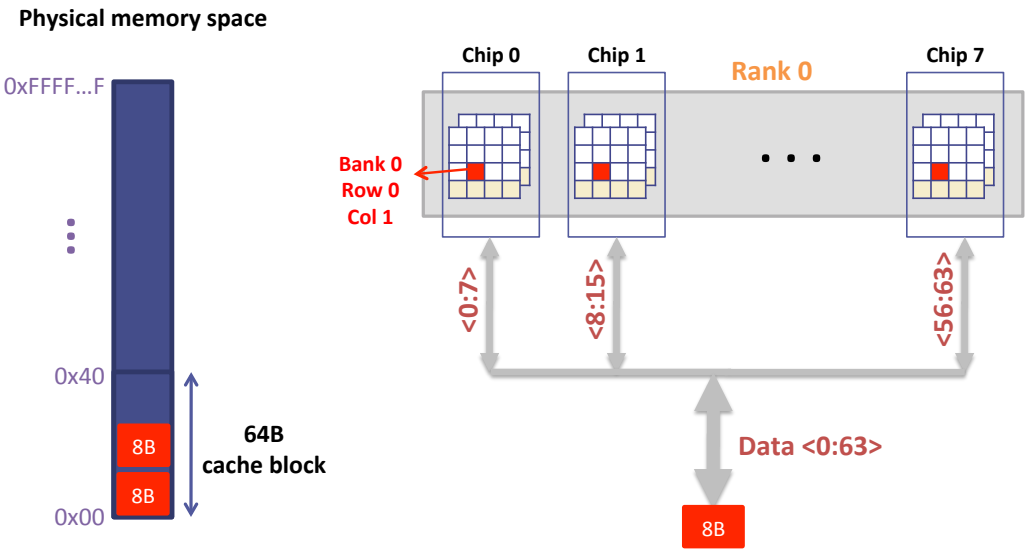
Example: Transferring a cache block



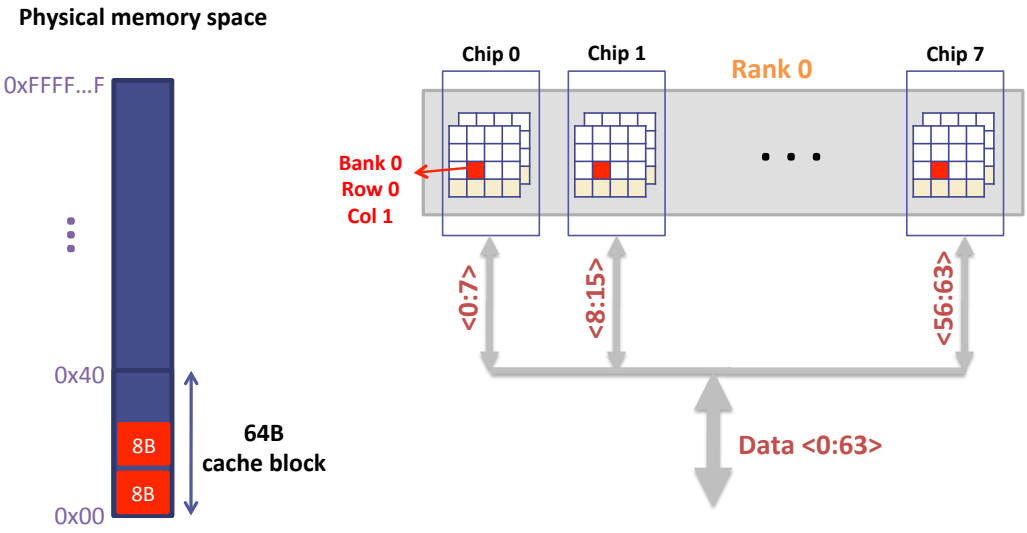
Example: Transferring a cache block



Example: Transferring a cache block



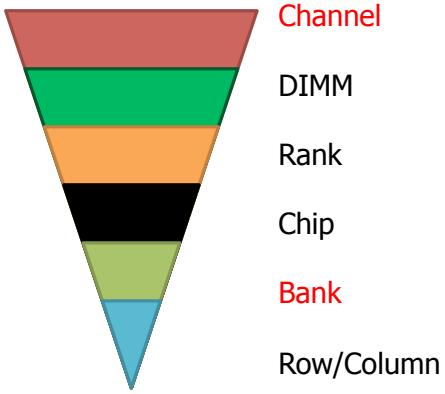
Example: Transferring a cache block



A 64B cache block takes 8 I/O cycles to transfer.
During the process, 8 columns are read sequentially.

Improve parallelism of memory access

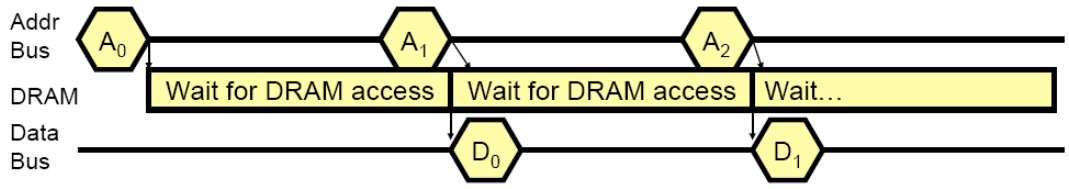
- Multiple banks
- Multiple channels



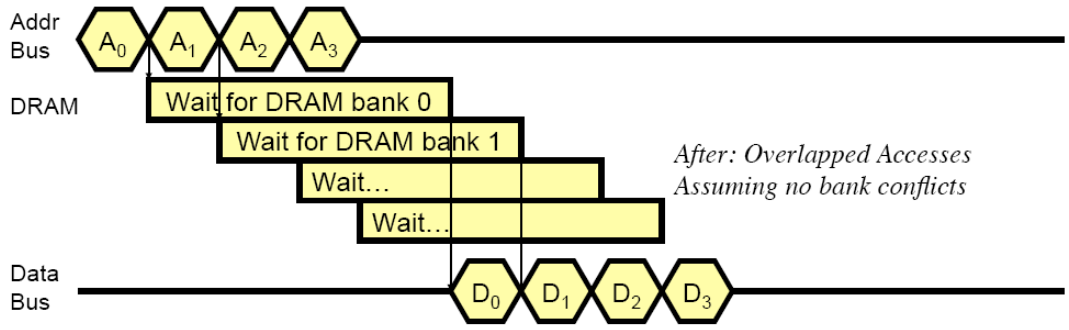
Memory Bank: A Fundamental Concept

- Why divide a chip into banks: Interleaving (banking)
  - Problem: a single monolithic memory array takes long to access and does not enable multiple accesses in parallel
  - Goal: Reduce the latency of memory array access and enable multiple accesses in parallel
  - Idea: Divide the array into multiple banks that can be accessed independently (in the same cycle or in consecutive cycles)
    - Each bank is smaller than the entire memory storage
    - Accesses to different banks can be overlapped
    - Can access different rows in different banks in parallel

How Multiple Banks Help



Before: No Overlapping
Assuming accesses to different DRAM rows



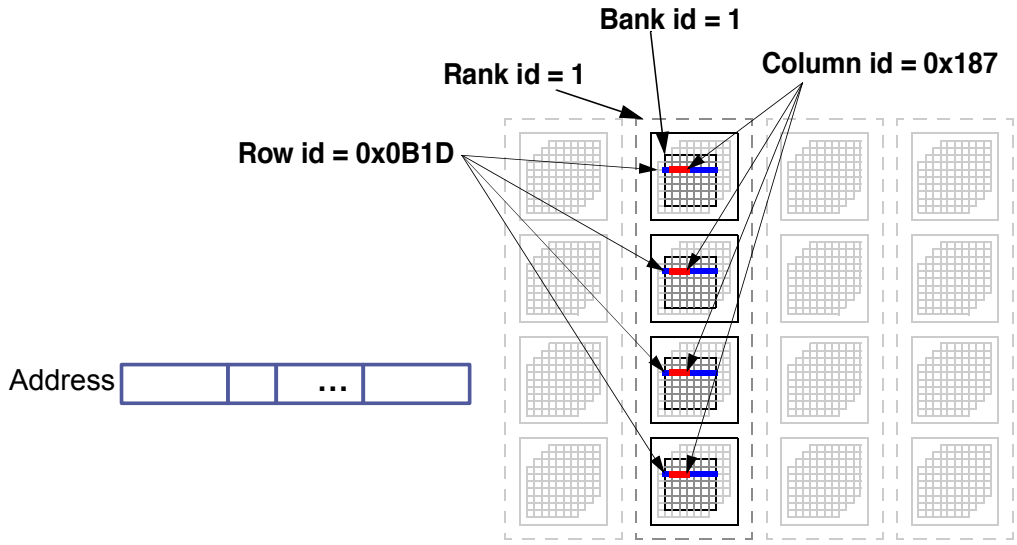
After: Overlapped Accesses
Assuming no bank conflicts

Multiple Channels: Even better

- Multiple banks
  - Enable concurrent DRAM accesses
  - Bits in address determine which bank an address resides in
- Multiple independent channels serve the same purpose
  - But they are even better because they have separate data buses
  - Increased bus bandwidth

- Enabling more concurrency requires reducing
  - Bank conflicts
  - Channel conflicts
- How to select/randomize bank/channel indices in address?
  - Lower order bits have more entropy
  - Randomizing hash functions (XOR of different address bits)

How do we encode the address: Address mapping



Example Address Mapping

- Single-channel system with 8-byte memory bus
  - 2GB memory, 8 banks, 16K rows & 2048 columns per bank, 64B cache blocks
- Row interleaving
  - Ensure consecutive rows of memory mapped to consecutive (i.e., different) banks

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

  - Accesses to consecutive rows serviced in a pipelined manner
- Cache block interleaving
  - Ensure consecutive cache block addresses mapped to consecutive banks

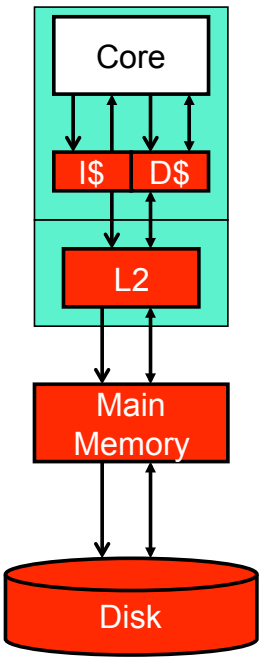
	8 bits		3 bits	
Row (14 bits)	High Column	Bank (3 bits)	Low Col.	Byte in bus (3 bits)

  - Accesses to consecutive cache blocks serviced in pipelined manner

Today

- Where is data – how to access main memory using an address
- Improve parallelism of memory access
- Address mapping

- Virtual memory

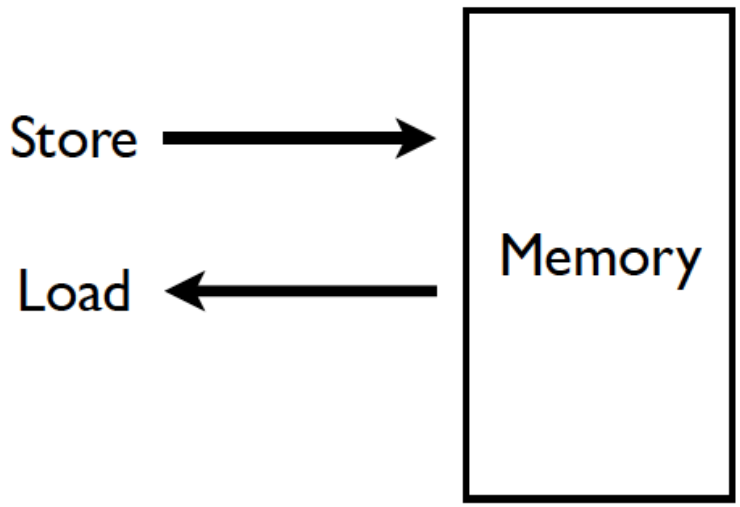


Virtualizing the memory hierarchy: An Infinite capacity

Ideal Memory

- Zero access time (latency)
- Infinite capacity
- Zero cost
- Infinite bandwidth (to support multiple accesses in parallel)

Memory (Programmer's View)



Abstraction: Virtual vs. Physical Memory

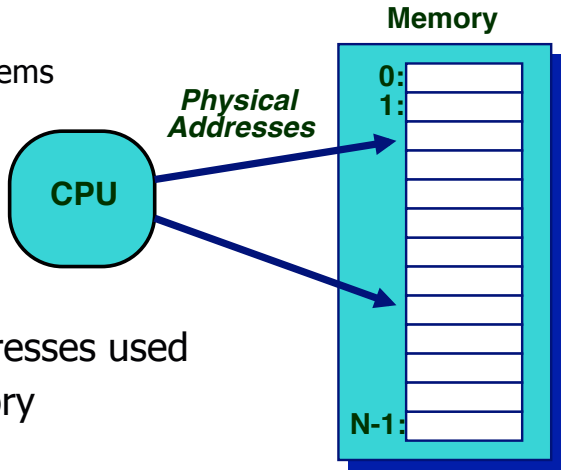
- Programmer sees virtual memory
  - Can assume the memory is "infinite"
- Reality: Physical memory size is much smaller than what the programmer assumes
- The system (system software + hardware, cooperatively) maps virtual memory addresses to physical memory
  - The system automatically manages the physical memory space transparently to the programmer

+ Programmer does not need to know the physical size of memory nor manage it → A small physical memory can appear as a huge one to the programmer → Life is easier for the programmer

-- More complex system software and architecture

A System with Physical Memory Only

- Examples:
  - most Cray machines
  - early PCs
  - nearly all embedded systems



CPU's load or store addresses used directly to access memory

The Problem

- Physical memory is of limited size (cost)
  - What if you need more?
  - Should the programmer be concerned about the size of code/data blocks fitting physical memory?
  - Should the programmer manage data movement from disk to physical memory?
  - Should the programmer ensure two processes do not use the same physical memory?

- Also, ISA can have an address space greater than the physical memory size
  - E.g., a 64-bit address space with byte addressability
  - What if you do not have enough physical memory?

Virtual Memory

- Idea: Give the programmer the illusion of a large address space while having a small physical memory
  - So that the programmer does not worry about managing physical memory

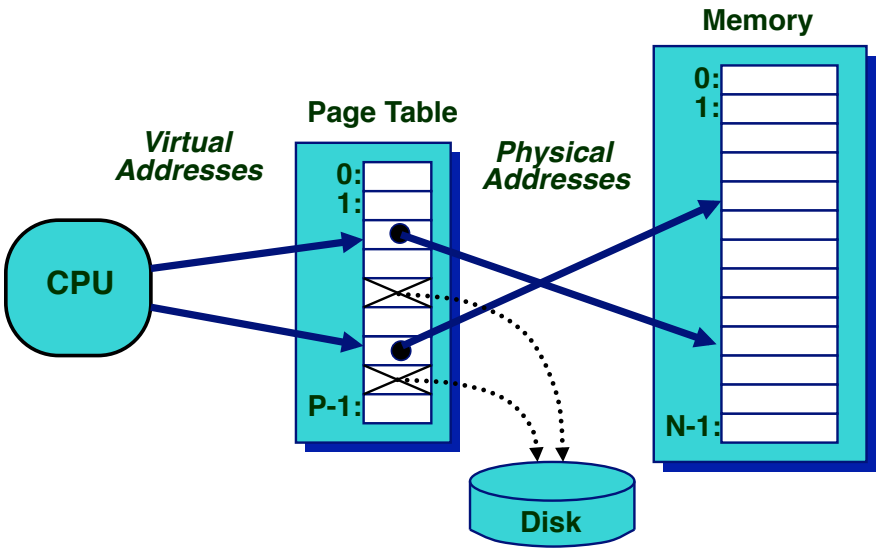
- Programmer can assume he/she has "infinite" amount of physical memory

- Hardware and software cooperatively and automatically manage the physical memory space to provide the illusion
  - Illusion is maintained for each independent process

Basic Mechanism

- Indirection (in addressing)
- Address generated by each instruction in a program is a "virtual address"
  - i.e., it is not the physical address used to address main memory
  - called "linear address" in x86
- An "address translation" mechanism maps this address to a "physical address"
  - called "real address" in x86
  - Address translation mechanism can be implemented in hardware and software together

A System with Virtual Memory (Page based)



- Address Translation: The hardware converts virtual addresses into physical addresses via an OS-managed lookup table (page table)