RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

Recitation 9

Jae Woo Joo

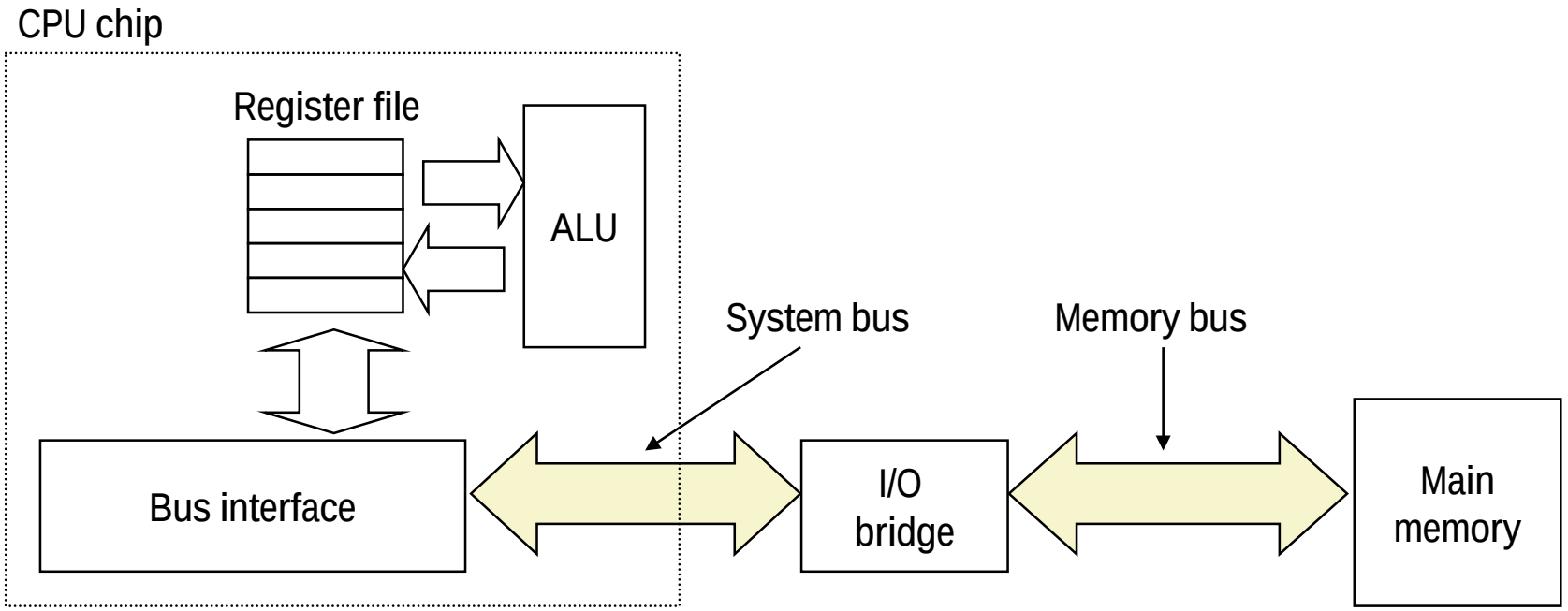
Random Access Memory (RAM)

- RAM comes in two varieties
  - Static RAM (SRAM)
    - Faster and more expensive than DRAM
    - Used for cache memories
  - Dynamic RAM (DRAM)
    - Used for main memory

- SRAM and DRAM are volatile memories
  - Lose information if powered off

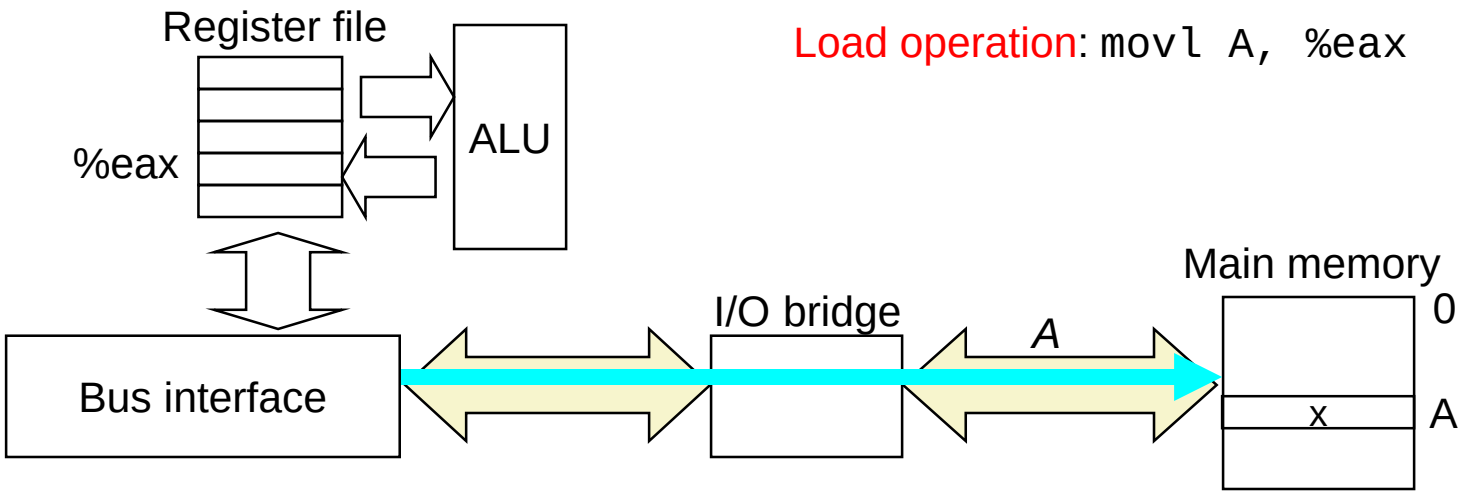
Bus Structure

- Bus: A collection of parallel wires that carry address, data, and control signals



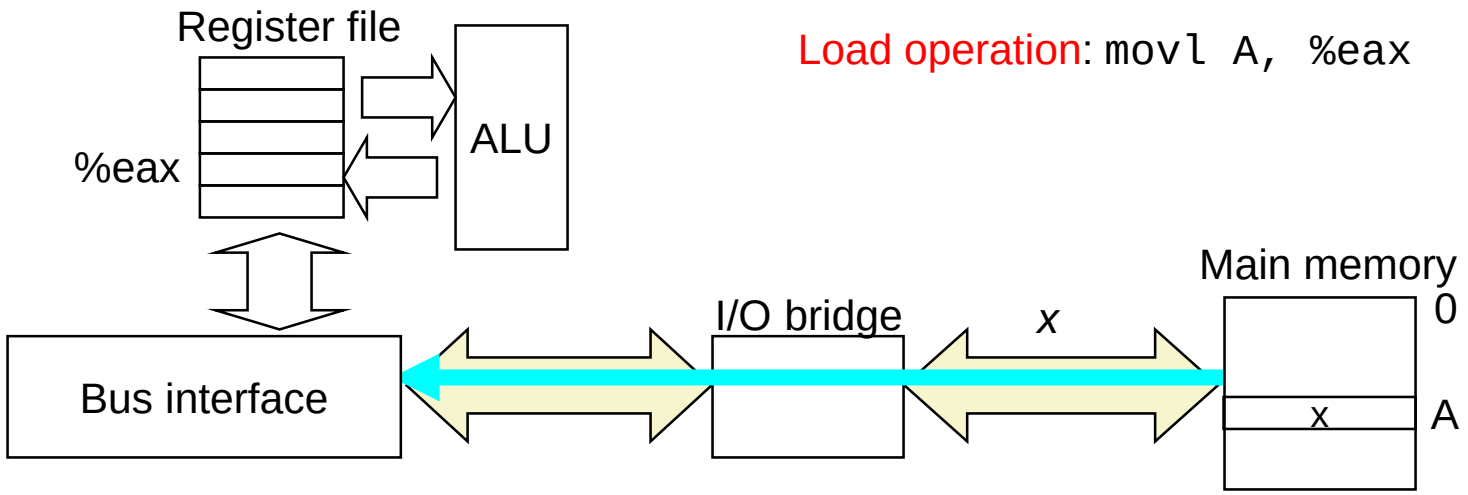
Memory Read Transaction (1)

- CPU places address A on the memory bus.



Memory Read Transaction (2)

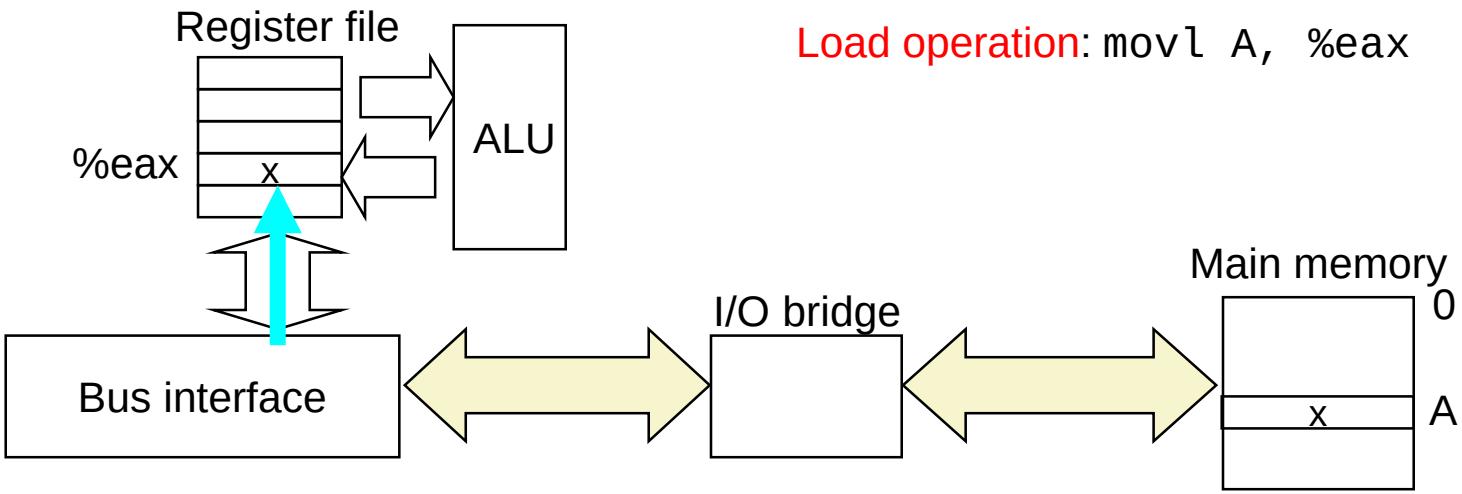
- Main memory reads A from the memory bus, retrieves word x, and places it on the bus.



Memory Read Transaction (3)

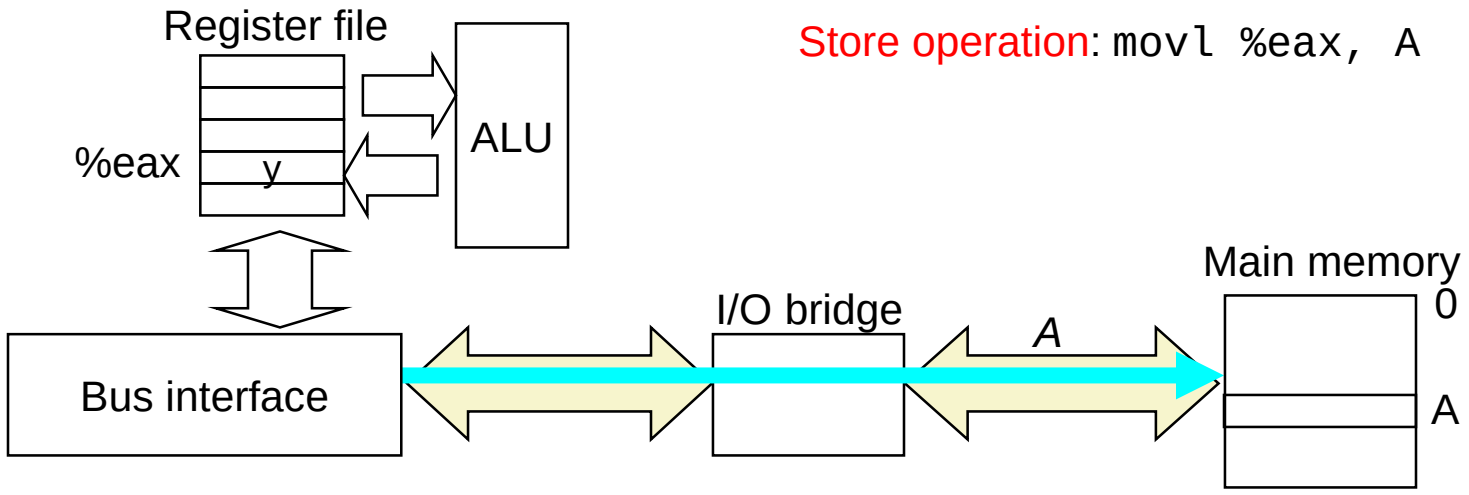
- CPU read word x from the bus and copies it into register %rax

.



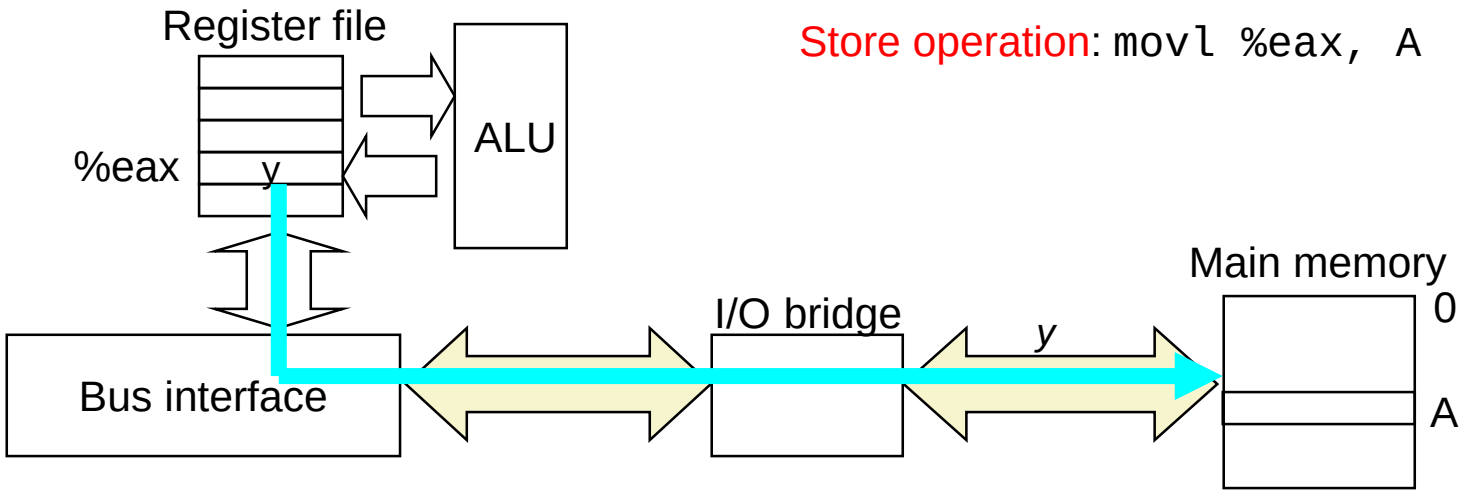
Memory Write Transaction (1)

- CPU places address A on bus. Main memory reads it and waits for the corresponding data word to arrive.



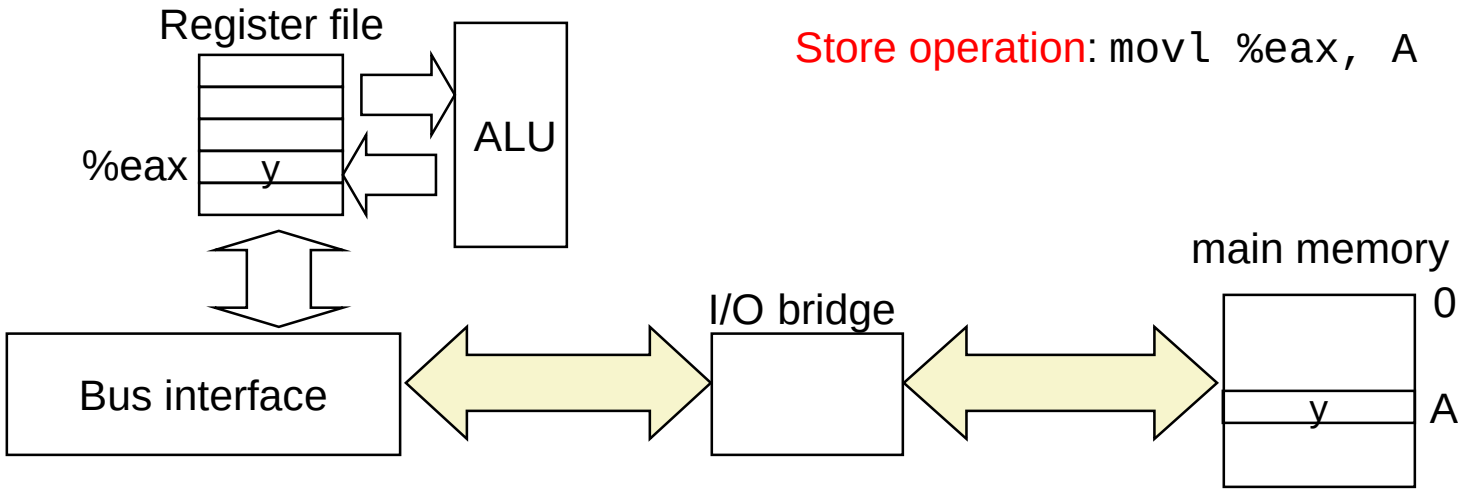
Memory Write Transaction (2)

- CPU places data word y on the bus.



Memory Write Transaction (3)

- Main memory reads data word y from the bus and stores it at address A.

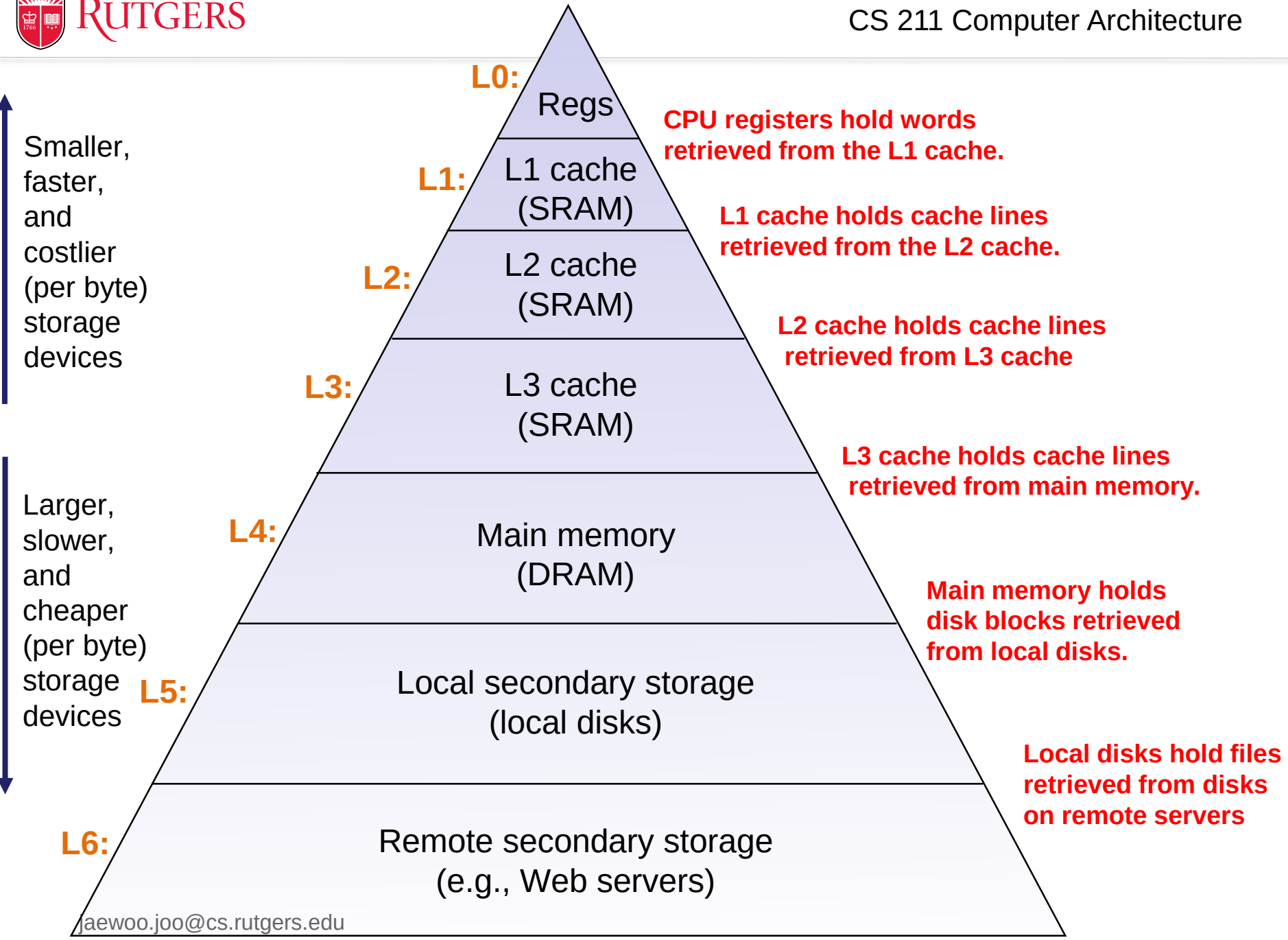


Locality

- Principle of Locality
  - Programs tend to use data and instructions with addresses near or equal to those they have used recently
- Temporal locality
  - Recently referenced items are likely to be referenced again in the near future
- Spatial locality
  - Items with nearby addresses tend to be referenced close together

Memory Hierarchies

- Some fundamental and enduring properties of hardware and software:
  - Fast storage technologies cost more per byte, have less capacity, and require more power
  - The gap between CPU and main memory speed is widening
  - Well-written programs tend to exhibit good locality
- These fundamental properties complement each other beautifully
- They suggest an approach for organizing memory and storage systems known as a memory hierarchy


L0:
Regs
CPU registers hold words retrieved from the L1 cache.
L1:
L1 cache (SRAM)
L1 cache holds cache lines retrieved from the L2 cache.
L2:
L2 cache (SRAM)
L2 cache holds cache lines retrieved from L3 cache
L3:
L3 cache (SRAM)
L3 cache holds cache lines retrieved from main memory.
L4:
Main memory (DRAM)
Main memory holds disk blocks retrieved from local disks.
L5:
Local secondary storage (local disks)
Local disks hold files retrieved from disks on remote servers
L6:
Remote secondary storage (e.g., Web servers)
Smaller, faster, and costlier (per byte) storage devices
Larger, slower, and cheaper (per byte) storage devices

Caches

- Cache: A smaller, faster storage device that acts as a staging area for a subset of the data in a larger, slower device

- Fundamental idea of a memory hierarchy
  - For each k, the faster, smaller device at level k serves as a cache for the larger, slower device at level k+1

Caches

- Why do memory hierarches work?
  - Because of locality, programs tend to access the data at level k more often than they access the data at level k+1
  - Thus, the storage at level k+1 can be slower, and thus larger and cheaper per bit

- Big Idea
  - The memory hierarchy creates a large pool of storage that costs as much as the cheap storage near the bottom, but that serves data to programs at the rate of the fast storage near the top

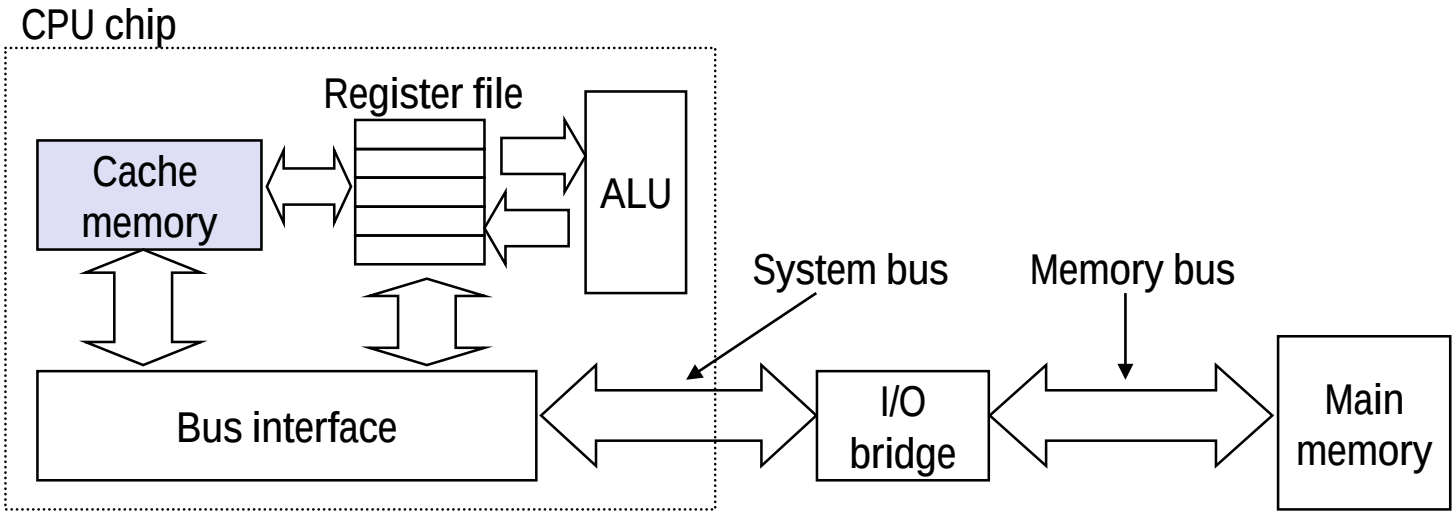
How does cache work?

- Temporal Locality (Locality in Time)
  - If an item is referenced, it will tend to be referenced again soon
  - Keep more recently accessed data items closer to the processor

- Spatial Locality (Locality in Space)
  - If an item is referenced, items whose addresses are close by tend to be referenced soon.
  - Move blocks consists of contiguous words to the cache

Cache Memories

- Cache memories are small, fast SRAM-based memories managed automatically in hardware
  - Hold frequently accessed blocks of main memory
- CPU looks first for data in cache



General Cache Concepts

- Cache Hits
  - When a program needs a particular data object d from level k+1, it first looks for d in one of the blocks currently stored at level k
  - If d happens to be cached at a level k, we call “cache hit”
- Cache Misses
  - If the data object d is not cached at level k, then we call “cache miss”

Cache Terminology

- Block
  - Minimum unit that may be cached
- Hit
  - Block is found in the cache
- Miss
  - Block is not found in the cache
- Miss ratio = (1 – Hit ratio)
  - Percent of misses compared to all accesses
- Hit time
  - Time to access the cache
- Miss penalty
  - Time to replace block on a miss

AMAT (Average Memory Access Time)

- Very powerful tool to estimate performance
- AMAT = Hit Time (HT) + Miss Rate (MR) * Miss Penalty (MP)
- Ex) If the cache has hit rate of 95%, hit time of 4 cycle, and a miss penalty of 100 cycle, what is the AMAT?

AMAT (Average Memory Access Time)

- Very powerful tool to estimate performance
- AMAT = Hit Time (HT) + Miss Rate (MR) * Miss Penalty (MP)
- Ex) If the cache has hit rate of 95%, hit time of 4 cycle, and a miss penalty of 100 cycle, what is the AMAT?
  - AMAT = 4 + 0.05 * 100 = 9 cycle

AMAT (Average Memory Access Time)

- AMAT = Hit Time (HT) + Miss Rate (MR) * Miss Penalty (MP)

- If we use L2 cache,
  - AMAT = HT(L1) + MR(L1) * MP(L1)
  - MP(L1) = HT(L2) + MR(L2) * MP(L2)

- Ex)
  - L1 cache hits in 1 cycle with hit rate 50%
  - L2 cache hits in 10 cycles with hit rate 75%
  - Main memory always hits in 100 cycles

AMAT (Average Memory Access Time)

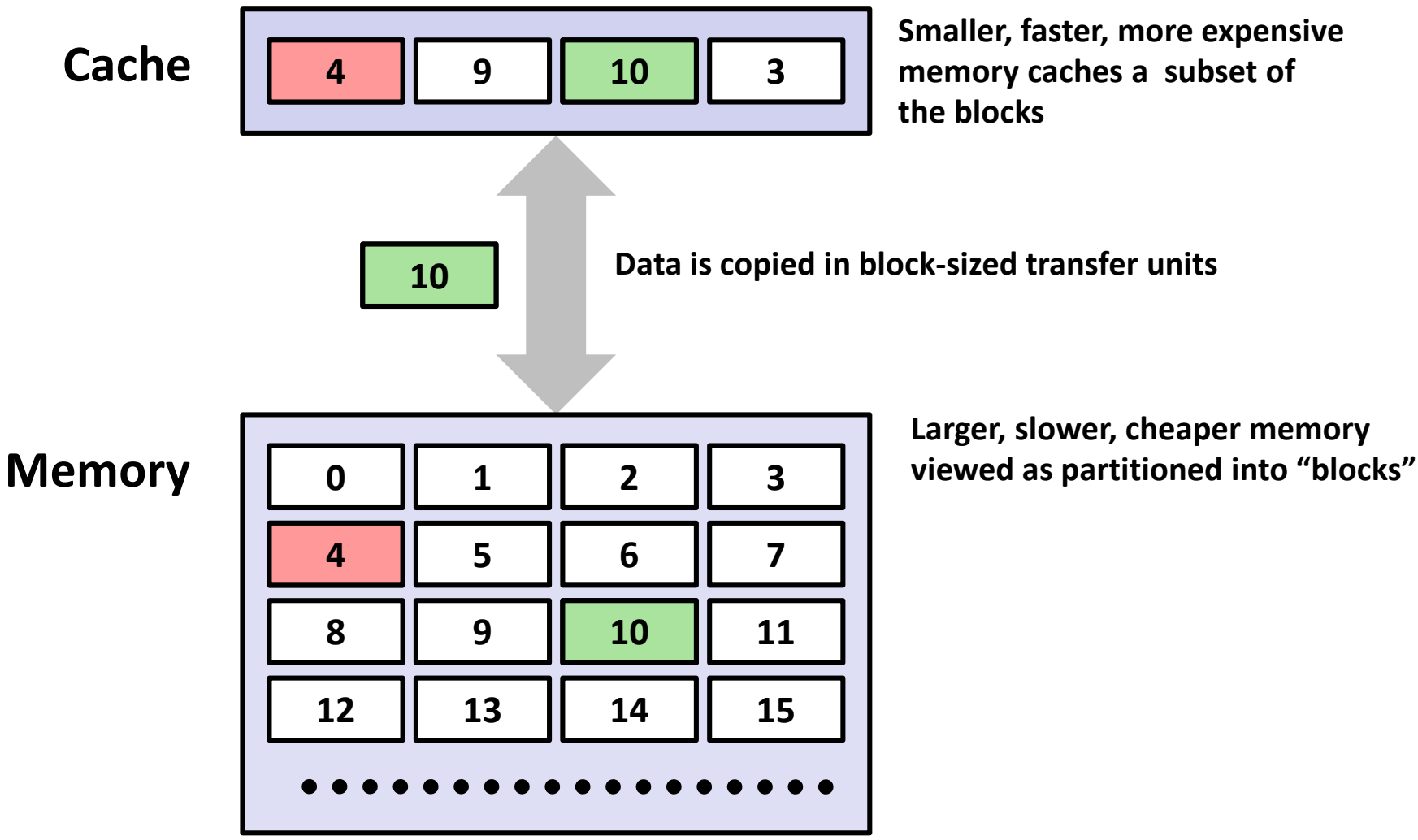
- AMAT = Hit Time (HT) + Miss Rate (MR) * Miss Penalty (MP)

- If we use L2 cache,
  - AMAT = HT(L1) + MR(L1) * MP(L1)
  - MP(L1) = HT(L2) + MR(L2) * MP(L2)

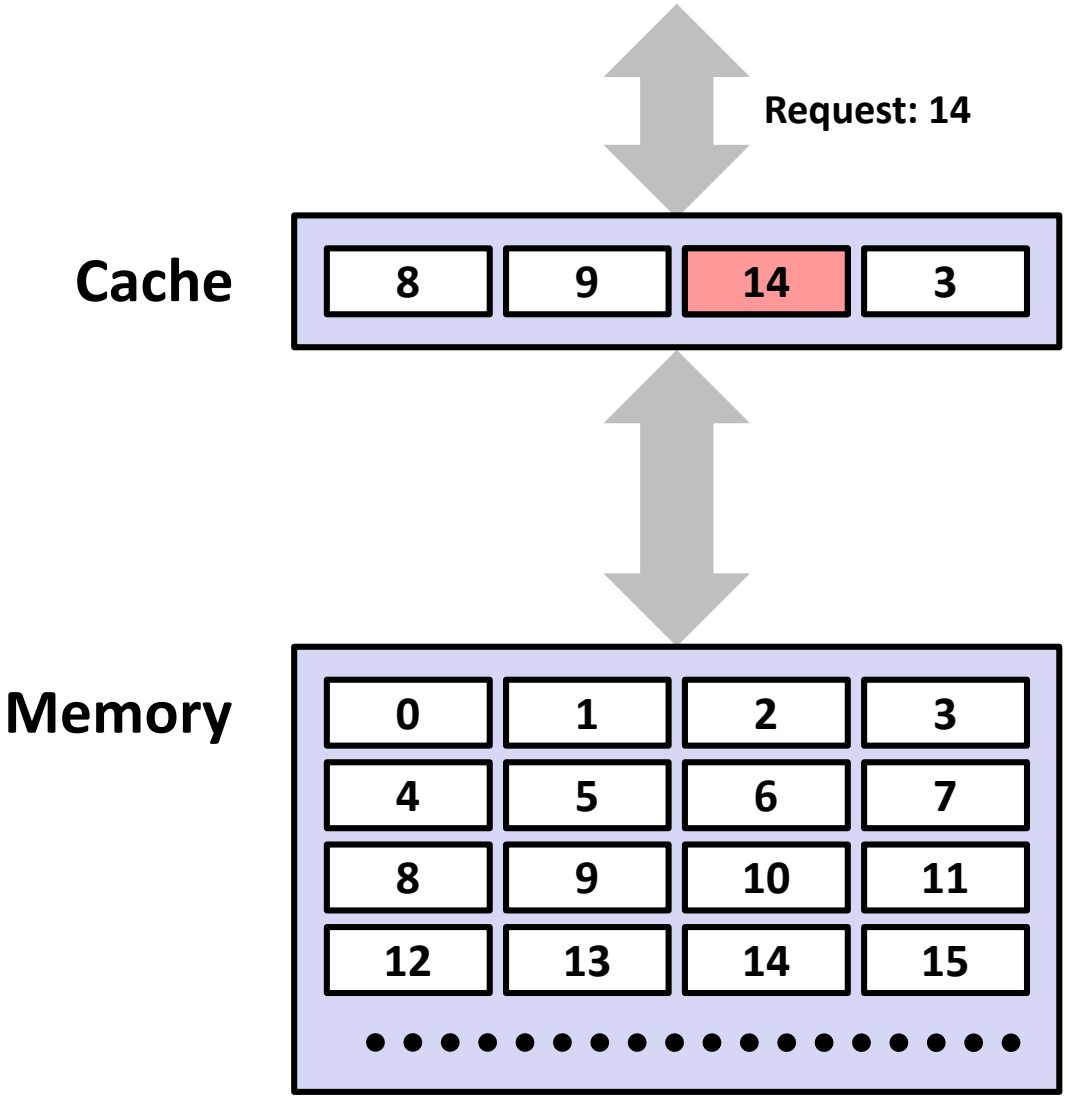
- Ex)
  - L1 cache hits in 1 cycle with hit rate 50%
  - L2 cache hits in 10 cycles with hit rate 75%
  - Main memory always hits in 100 cycles

  - AMAT = 1+ 0.5*(10+0.25*100) = 18.5 cycles

General Cache Concepts



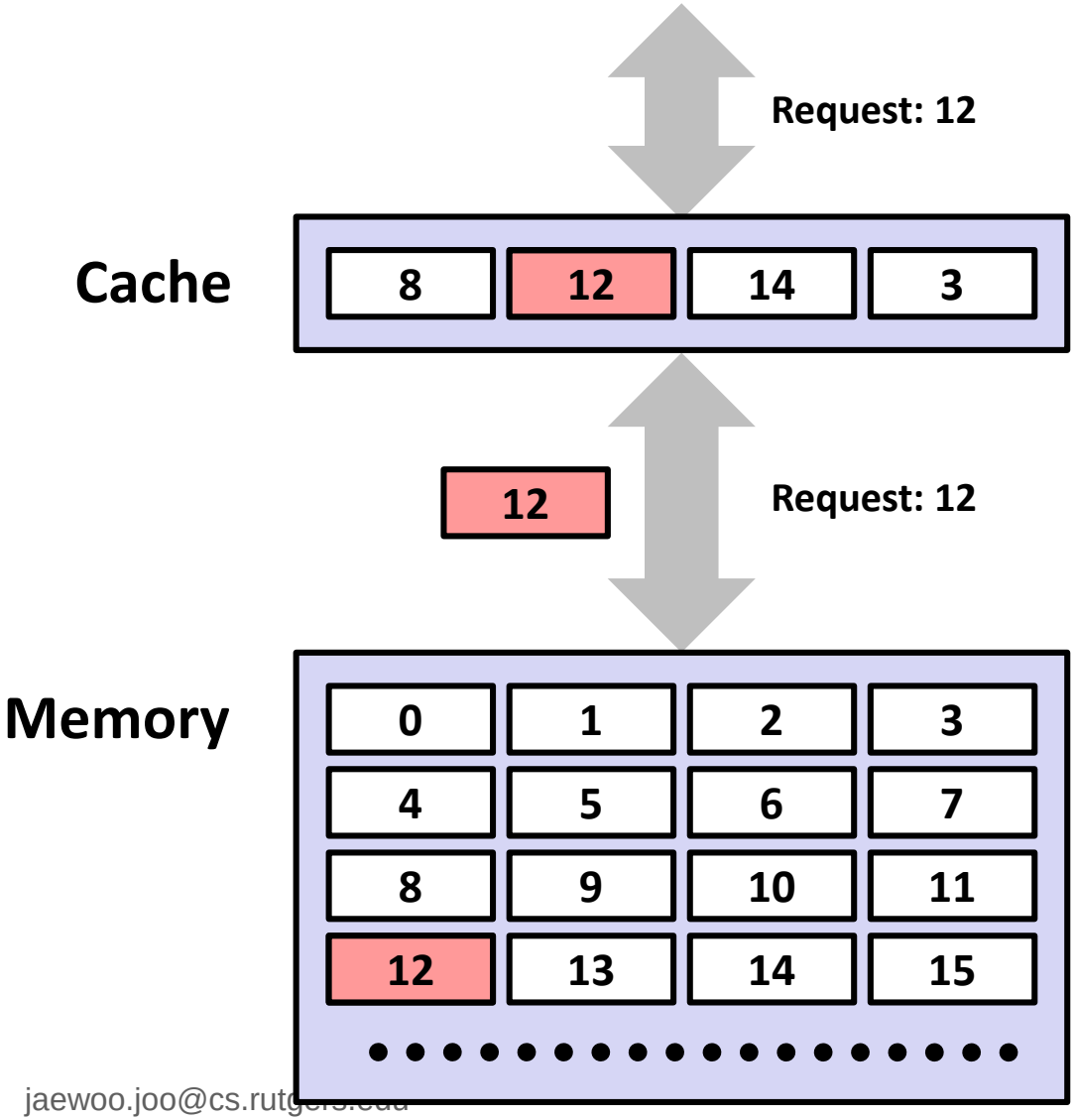
General Cache Concepts: Hit



Data in block b is needed

Block b is in cache:
Hit!

General Cache Concepts: Miss



Data in block b is needed

Block b is not in cache:
Miss!

Block b is fetched from memory

Block b is stored in cache

- Placement policy: determines where b goes
- Replacement policy: determines which block gets evicted (victim)

Q & A

- Any questions?