Lecture 14: Cache wrap-up and Memory Intro

- Last Time:
  - Replacement and write policies
  - Cache performance optimizations
- Today
  - Some more cache performance optimizations
  - Introduction to memories and addressing

Cache Review

- Locality: Spatial and Temporal
- Terms:
  - block size, associativity
  - Miss types: compulsory, capacity, conflict
- Improving performance
  - AMAT = $P_{hit}*T_{hit} + P_{miss}*T_{miss}$

Reducing Miss Rate: Increase Associativity

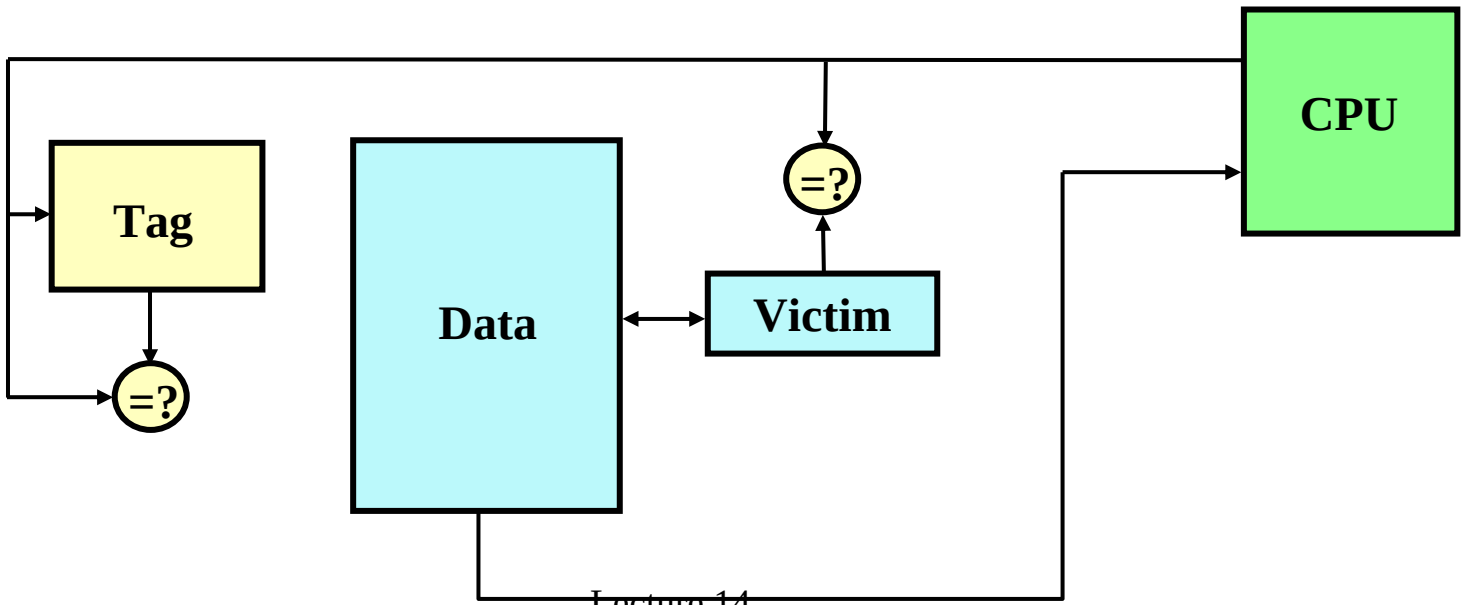
- Reduce conflict misses
- Rules of thumb
  - 8-way = fully associative
  - Direct mapped size N = 2-way set associative size N/2

- But!
  - Size N associative is larger than Size N direct mapped
  - Associative typically slower that direct mapped (t_{hit} larger)

Reducing Miss Rate: Use a "Victim" Cache

- Small cache (< 8 entries)
  - Jouppi 1990
  - Accessed in parallel with main cache
  - Captures conflict misses

Set	1W	2W	3W	4W	
0	X				
1	X	X			
2	X				
3	X	X	X	X	X
4	X				
5	X				
6					
7	X	X			



Reducing Miss Rate: Prefetching

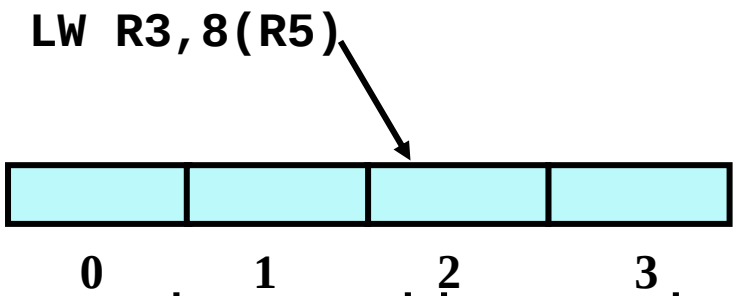
- Fetching Data that you will probably need

- Instructions
  - Alpha 21064 on cache miss
    - Fetches requested block intro instruction stream buffer
    - Fetches next sequential block into cache
- Data
  - Automatically fetch data into cache (spatial locality)
  - Issues?

- Compiler controlled prefetching
  - Inserts prefetching instructions to fetch data for later use
  - Registers or cache

Reduce Miss Penalty: Deliver Critical word first

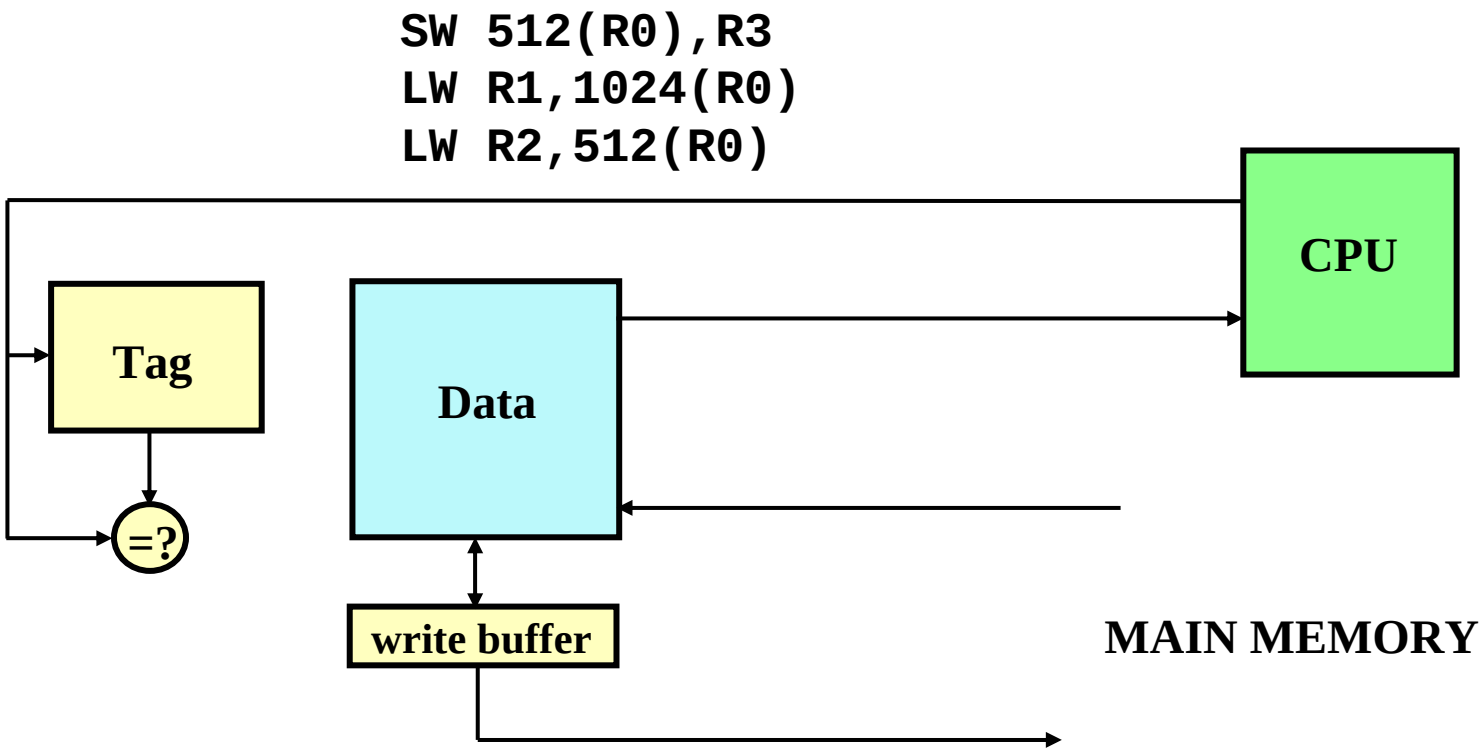
- Only need one word from block immediately



- Don't write entire word into cache first
  - Fetch word 2 first (deliver to CPU)
  - Fetch order: 2 3 0 1

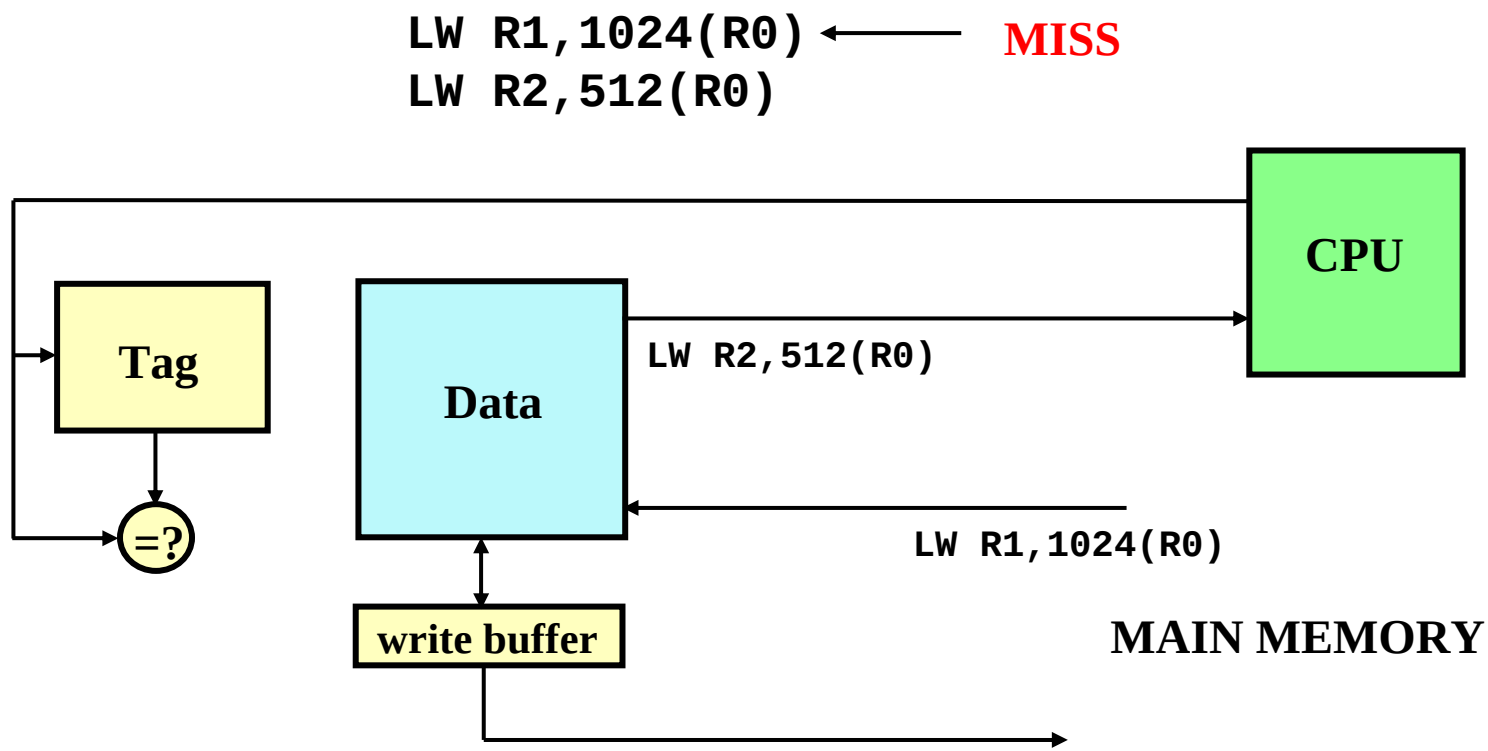
Reduce Miss Penalty: Read Misses First

- Let reads pass writes in Write buffer



Reduce Miss Penalty: Lockup Free Cache

- Let cache continue to function while miss is being serviced

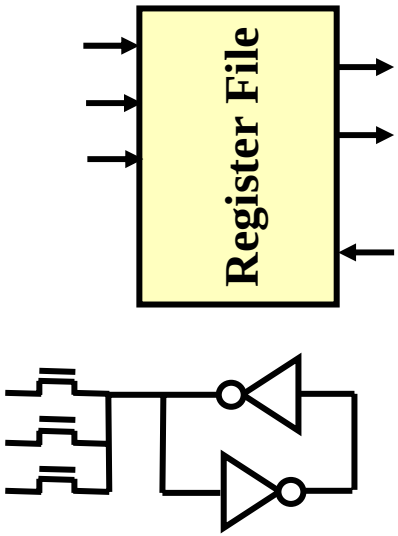


Memory systems

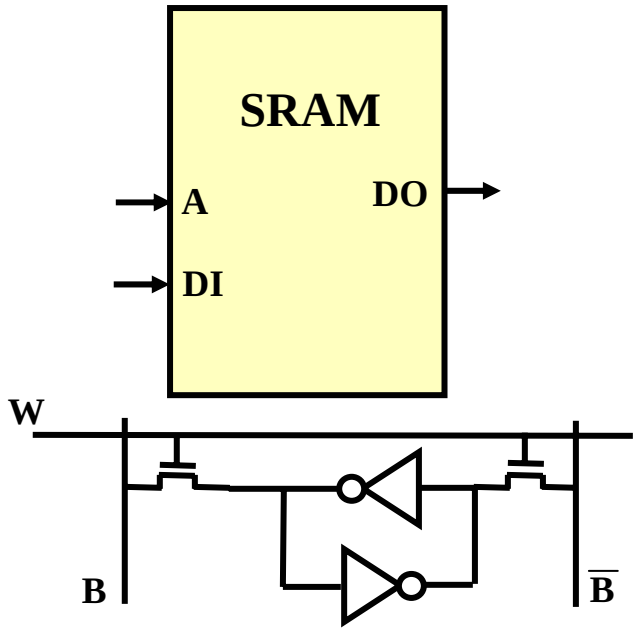
Memory Systems

- Memory Technology
  - SRAM, DRAM
- Higher order memory functions
  - Relocation, protection
- Virtual memory

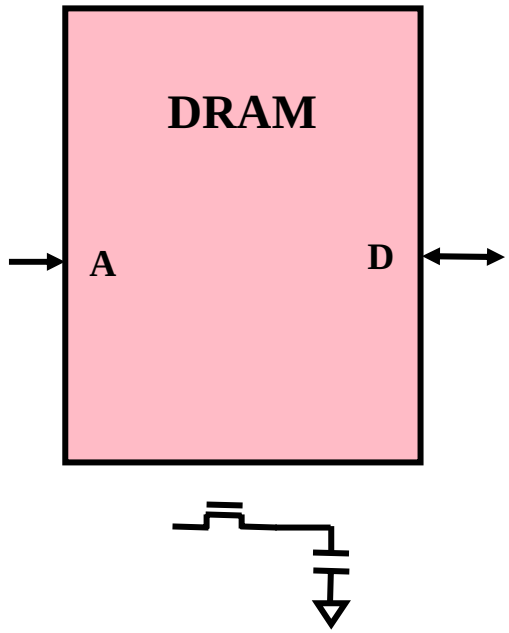
Typical Memory Technologies



- Integrated into CPU
- Fast, many ports



- Caches
  - On chip L1
  - On/off chip L2
  - Off chip L3
- More bits, fewer ports

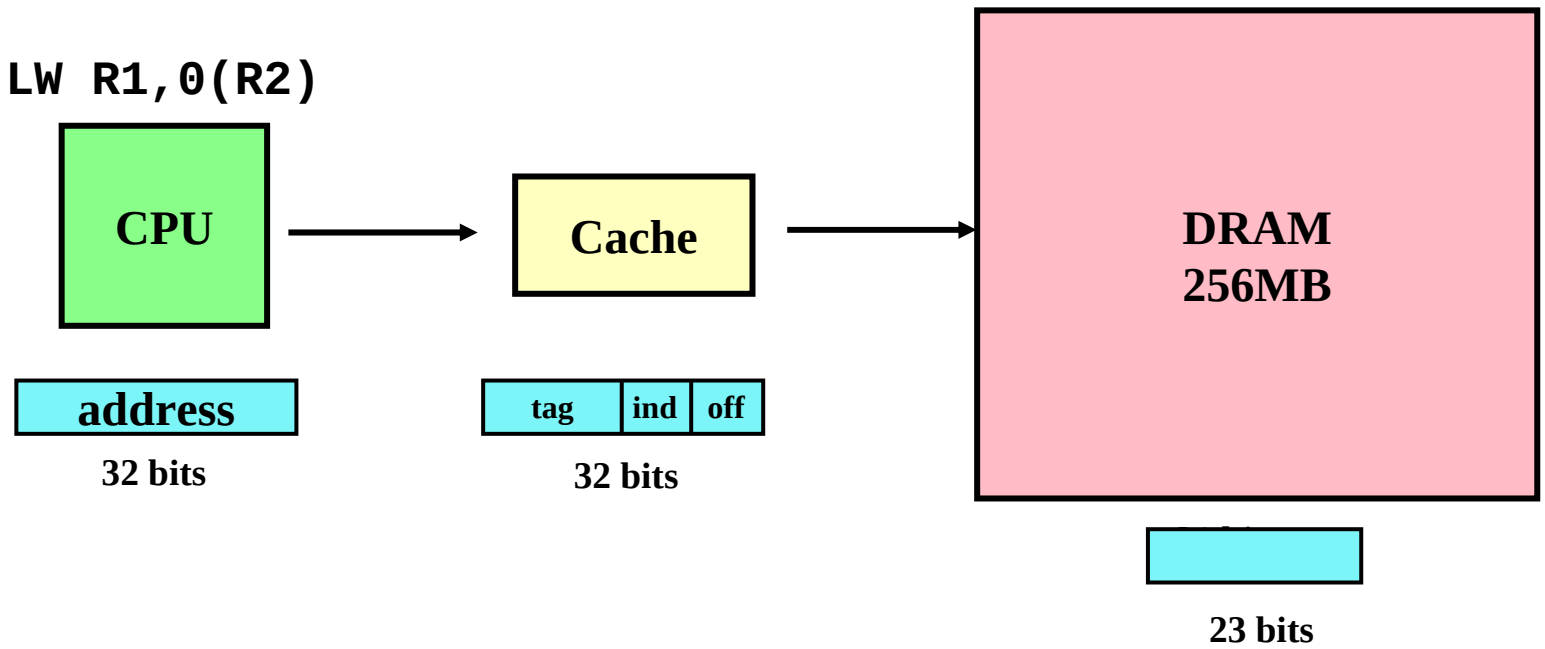


- Main Memory
- Very dense
- Slower to access, one port

SRAM vs. DRAM

- SRAM
  - 4MBit capacity
  - 64-bit data interface
  - 15-20ns access time
    - (50-80MHz)
  - Storage cells are self-restoring
  - Lower power
  - 4 times cost per bit (vs DRAM)
- DRAM
  - 512Mbit capacity
  - 16-bit interfaces
  - 45ns read access time
    - 22MHz
  - Reads destroy data
    - Must write data back
    - Refresh periodically
  - Higher power
  - Lower cost per bit

Physical Memory Addressing

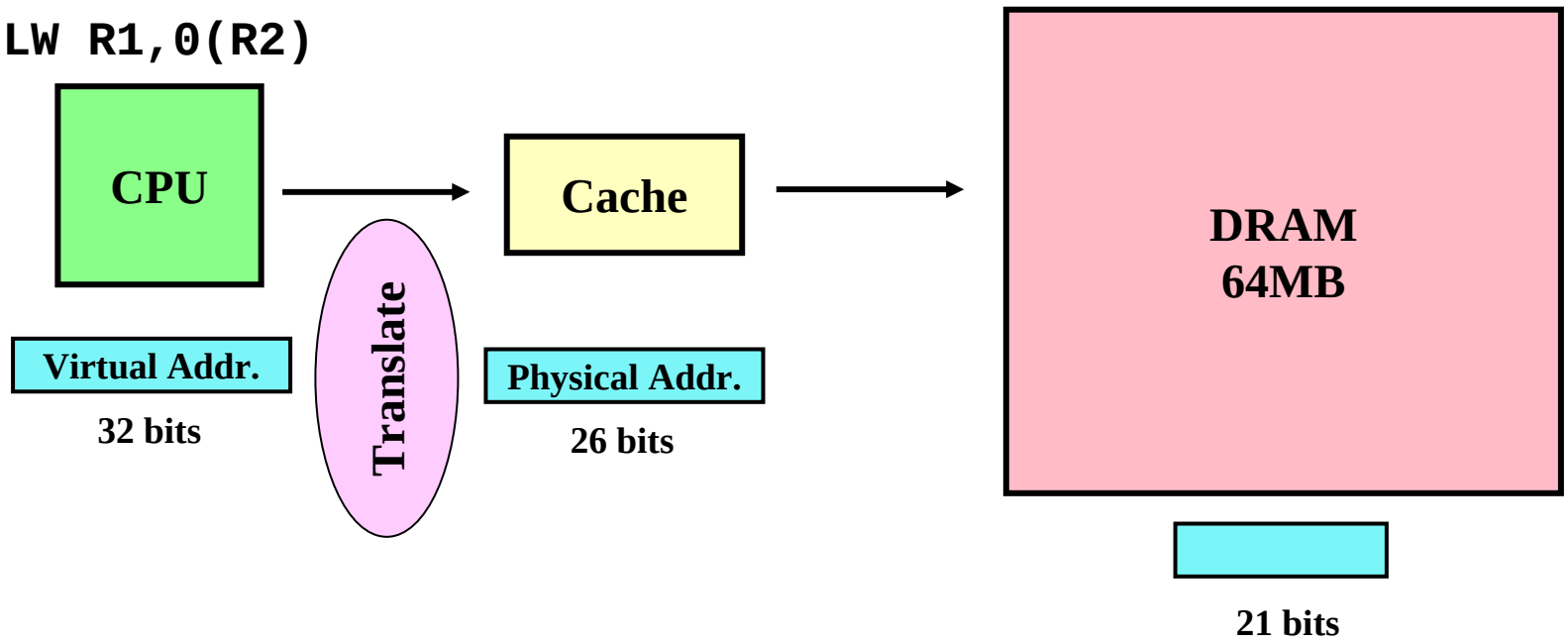


- **assuming 32 bytes per cache block**

What if?

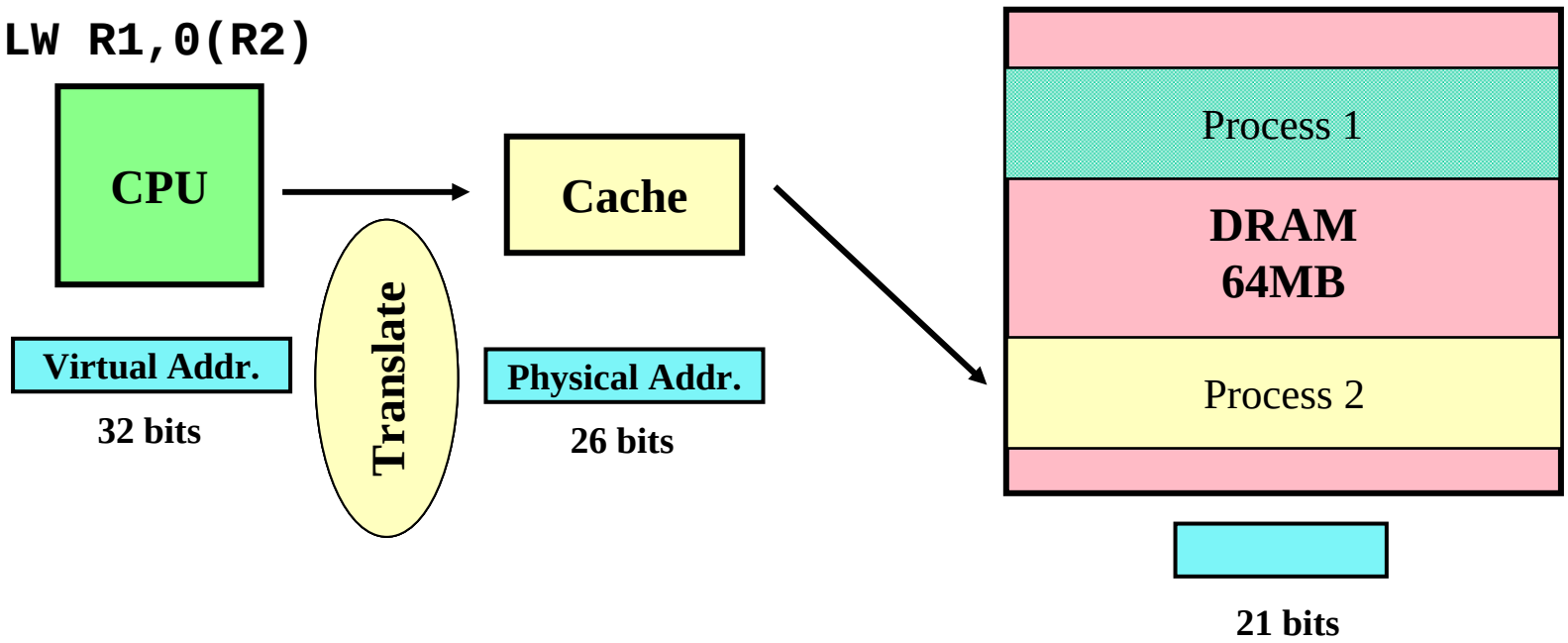
- A program is loaded into different places in memory each time it runs?
  - Relocation
- A program wants to use more memory than physically exists?
  - Page to disk
- We want to switch between multiple programs that use different data?
  - Protection

A Load to Virtual Memory



- Translate from virtual space to physical space
  - VA $\Rightarrow$ PA
  - May need to go to disk

A Load to Virtual Memory



- Both programs can use the same set of addresses!
  - Change translation tables to point same VA to different PA for different programs

Summary

- More cache performance optimizations
- Virtual memory provides
  - Illusion of private memory system for each process
  - Protection
  - Relocation in memory system
  - Demand paging
- But – page tables can be large
  - Motivates: paging page tables, multi-level tables, inverted page tables

- Next time
  - Exam #1, covers up through section 7.2 in the book