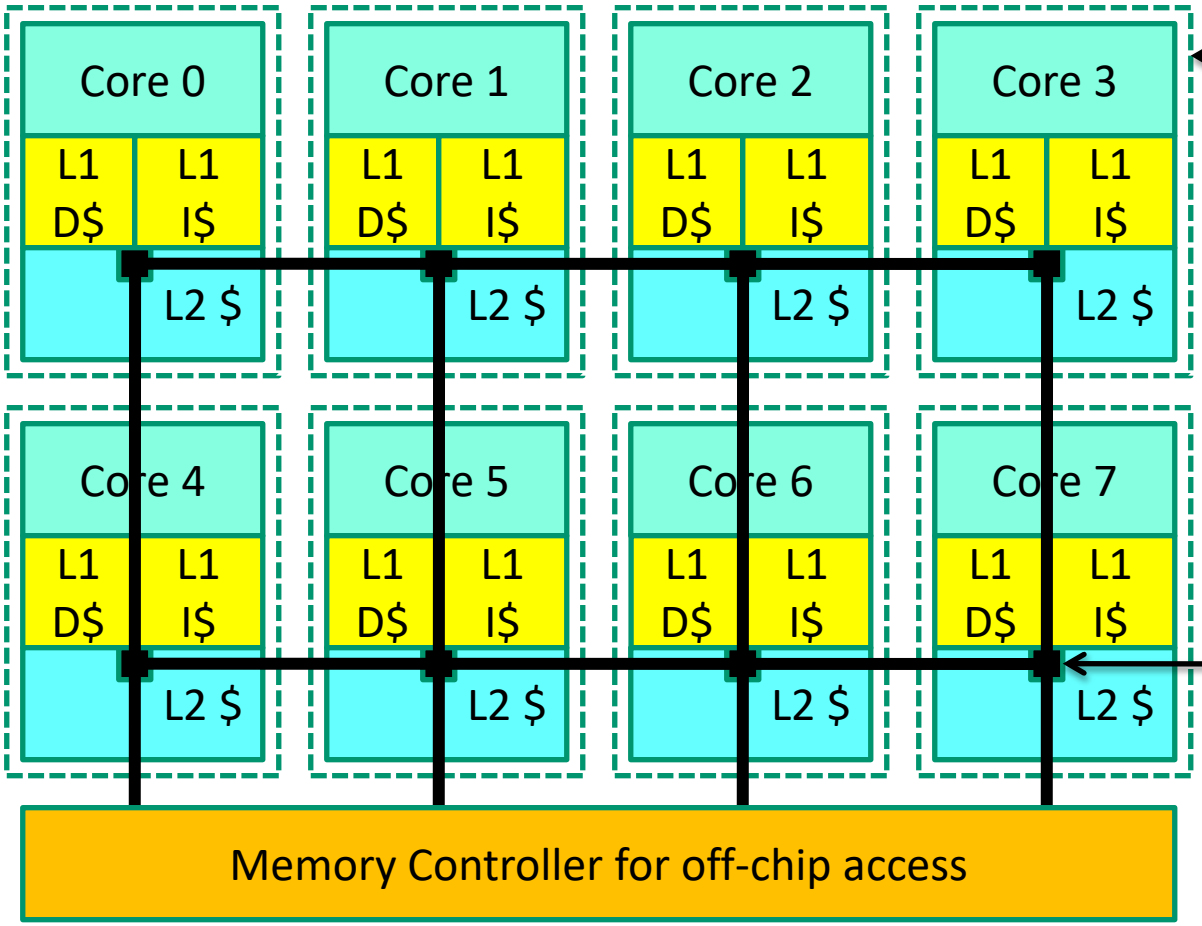
Lecture: Virtual Memory

- Topics: virtual memory details, TLB/cache access, superpages

Shared NUCA Cache



A single tile composed of a core, L1 caches, and a bank (slice) of the shared L2 cache

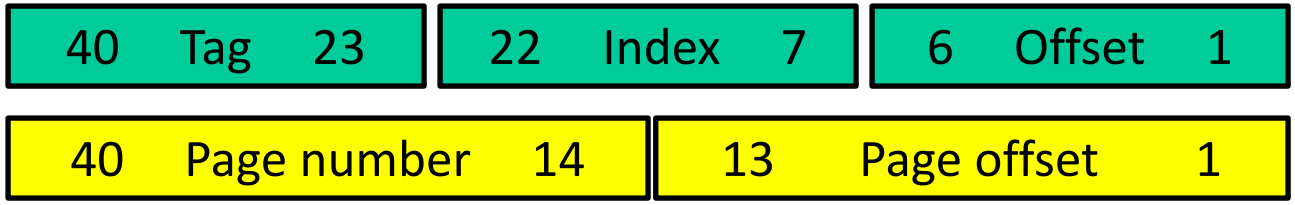
The cache controller forwards address requests to the appropriate L2 bank and handles coherence operations

Problem 1

- Assume a large shared LLC that is tiled and distributed on the chip. Assume 16 tiles. Assume an OS page size of 8KB. The entire LLC has a size of 32 MB, uses 64-byte blocks, and is 8-way set-associative. Which of the 40 physical address bits are used to specify the tile number? Provide an example page number that is assigned to tile 0.

Problem 1

- Assume a large shared LLC that is tiled and distributed on the chip. Assume 16 tiles. Assume an OS page size of 8KB. The entire LLC has a size of 32 MB, uses 64-byte blocks, and is 8-way set-associative. Which of the 40 physical address bits are used to specify the tile number? Provide an example page number that is assigned to tile 0.

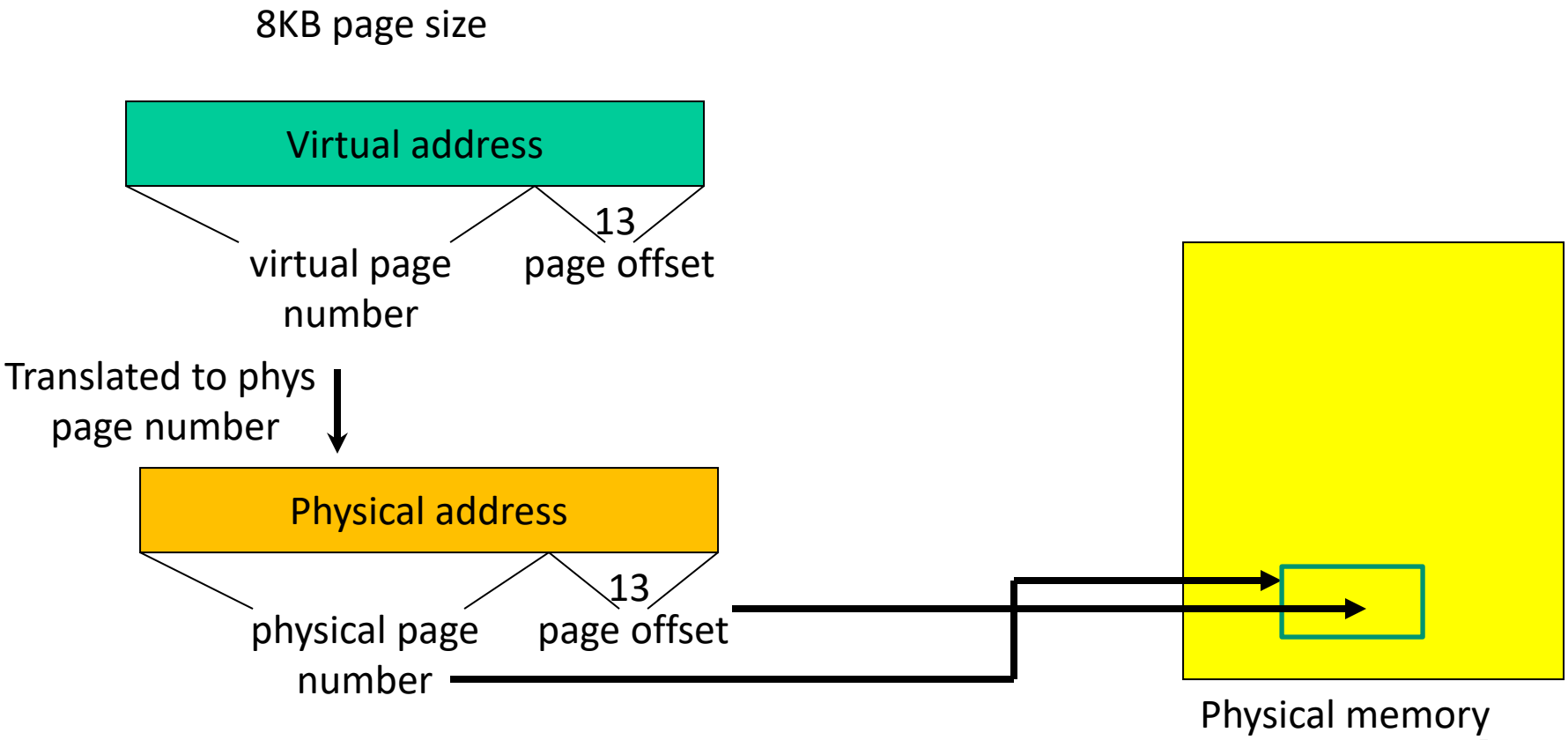


The cache has 64K sets, i.e., 6 block offset bits, 16 index bits, and 18 tag bits. The address also has a 13-bit page offset, and 27 page number bits. Nine bits (bits 14-22) are used for the page number and the index bits. Any four of those bits can be used to designate the tile number, say, bits 19-22. An example page number assigned to tile 0 is xxx…xxx0000xxx…xxx

bit 22 19

Address Translation

- The virtual and physical memory are broken up into pages

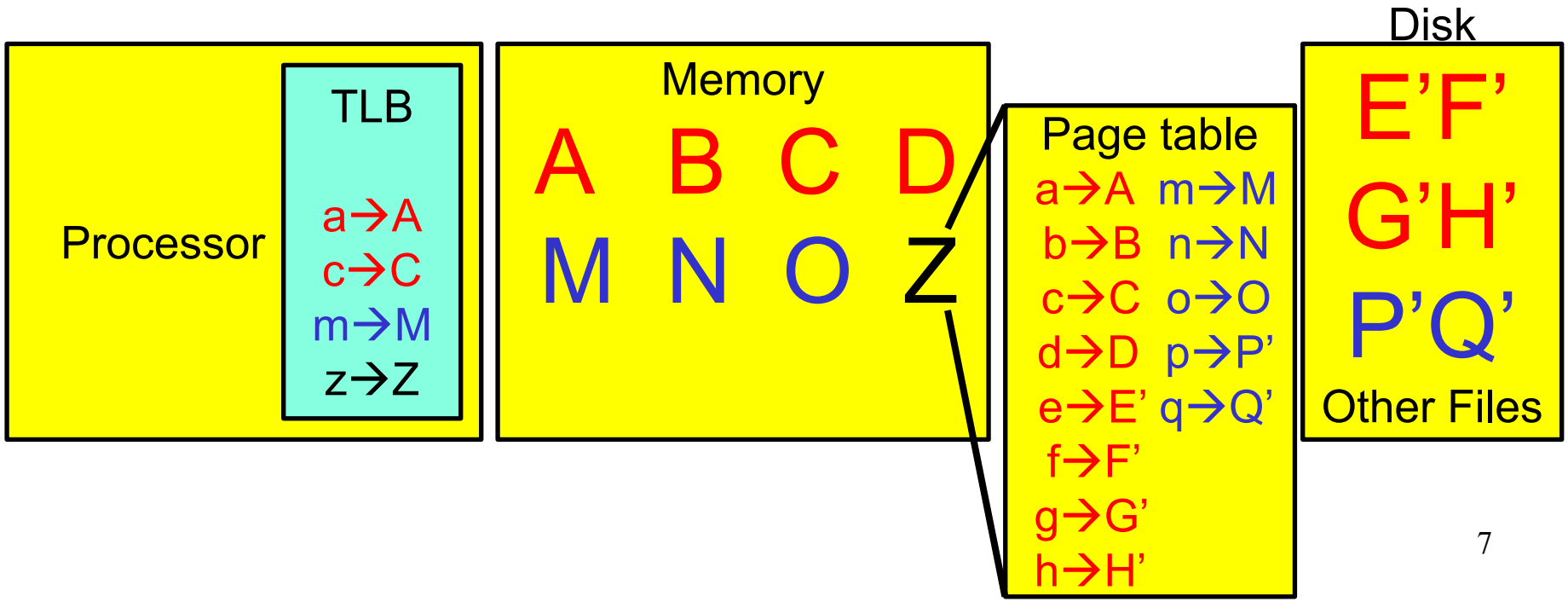


TLB

- Since the number of pages is very high, the page table capacity is too large to fit on chip
- A translation lookaside buffer (TLB) caches the virtual to physical page number translation for recent accesses
- A TLB miss requires us to access the page table, which may not even be found in the cache – two expensive memory look-ups to access one word of data!
- A large page size can increase the coverage of the TLB and reduce the capacity of the page table, but also increases memory waste

Problem 2

- Build an example toy virtual memory system. Each program has 8 virtual pages. Two programs are running together. The physical memory can store 8 total pages. Show example contents of the physical memory, disk, page table, TLB. Assume that virtual pages take names a-z and physical pages take names A-Z.



TLB and Cache

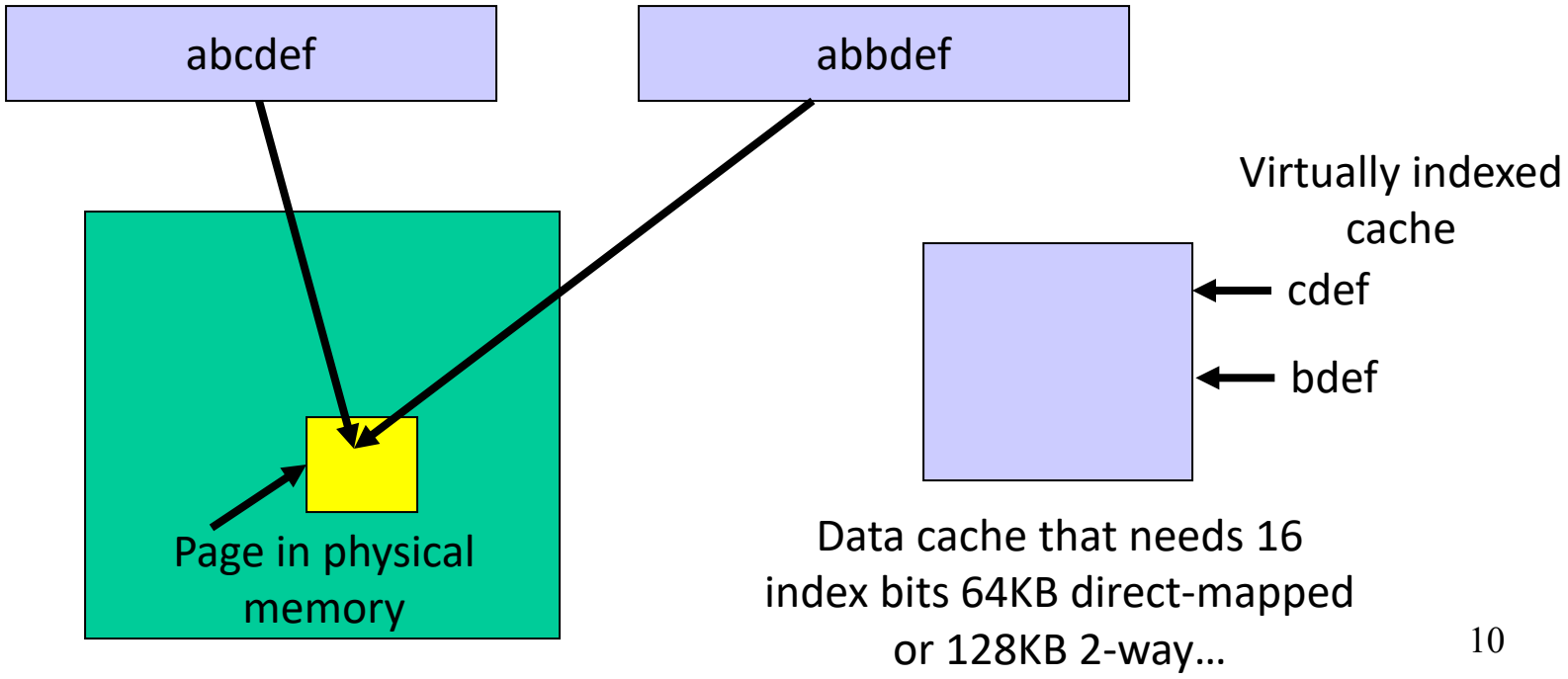
- Is the cache indexed with virtual or physical address?
  - To index with a physical address, we will have to first look up the TLB, then the cache → longer access time
  - Multiple virtual addresses can map to the same physical address – can we ensure that these different virtual addresses will map to the same location in cache? Else, there will be two different copies of the same physical memory word

- Does the tag array store virtual or physical addresses?
  - Since multiple virtual addresses can map to the same physical address, a virtual tag comparison can flag a miss even if the correct physical memory word is present

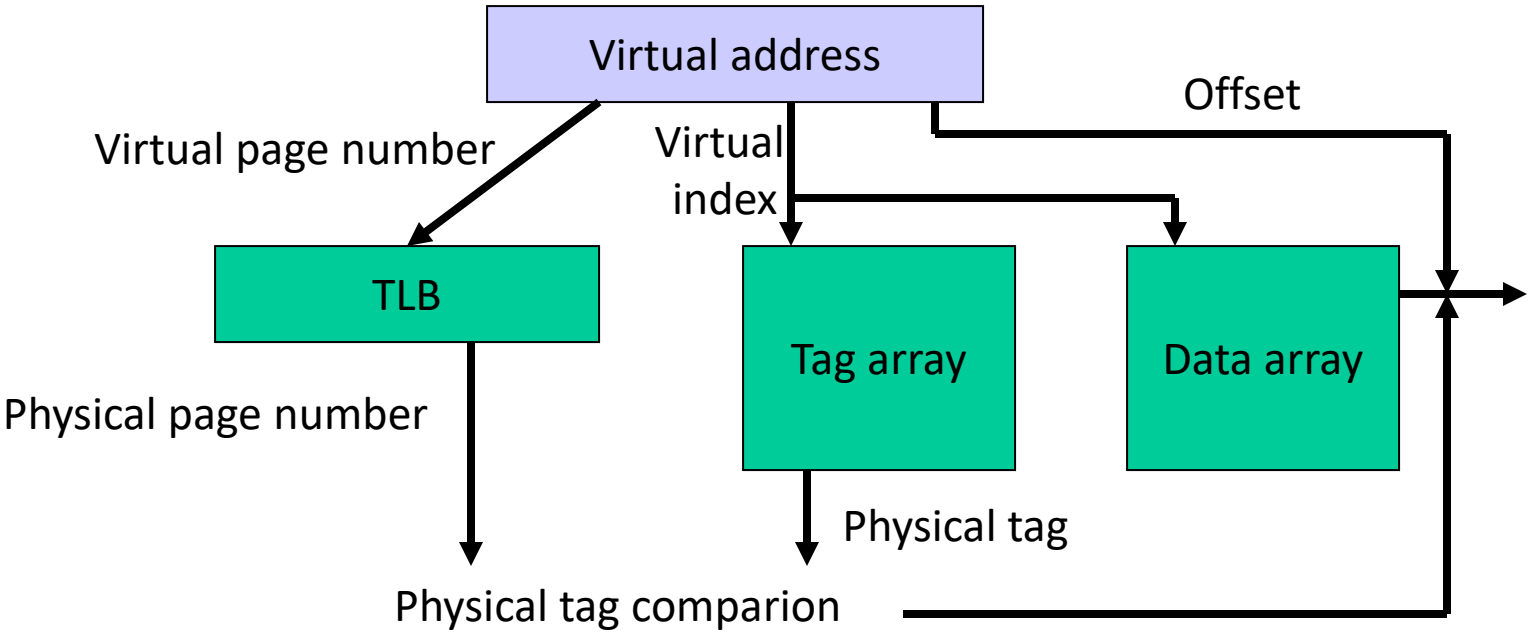
TLB and Cache

Virtually Indexed Caches

- 24-bit virtual address, 4KB page size → 12 bits offset and 12 bits virtual page number
- To handle the example below, the cache must be designed to use only 12 index bits – for example, make the 64KB cache 16-way
- Page coloring can ensure that some bits of virtual and physical address match



Cache and TLB Pipeline



Virtually Indexed; Physically Tagged Cache

Problem 3

- Assume that page size is 16KB and cache block size is 32 B. If I want to implement a virtually indexed physically tagged L1 cache, what is the largest direct-mapped L1 that I can implement? What is the largest 2-way cache that I can implement?

Problem 3

- Assume that page size is 16KB and cache block size is 32 B. If I want to implement a virtually indexed physically tagged L1 cache, what is the largest direct-mapped L1 that I can implement? What is the largest 2-way cache that I can implement?

  There are 14 page offset bits. If 5 of them are used for block offset, there are 9 more that I can use for index.

  512 sets ➔ 16KB direct-mapped or 32KB 2-way cache

Protection

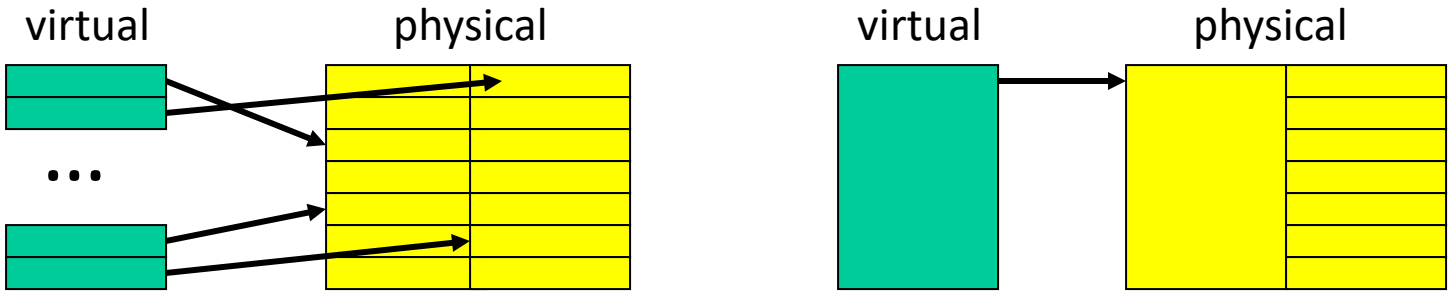
- The hardware and operating system must co-operate to ensure that different processes do not modify each other's memory
- The hardware provides special registers that can be read in user mode, but only modified by instrs in supervisor mode
- A simple solution: the physical memory is divided between processes in contiguous chunks by the OS and the bounds are stored in special registers – the hardware checks every program access to ensure it is within bounds
- Protection bits are tracked in the TLB on a per-page basis

Superpages

- If a program's working set size is 16 MB and page size is 8KB, there are 2K frequently accessed pages – a 128-entry TLB will not suffice
- By increasing page size to 128KB, TLB misses will be eliminated – disadvantage: memory waste, increase in page fault penalty
- Can we change page size at run-time?
- Note that a single page has to be contiguous in physical memory

Superpages Implementation

- At run-time, build superpages if you find that contiguous virtual pages are being accessed at the same time
- For example, virtual pages 64-79 may be frequently accessed – coalesce these pages into a single superpage of size 128KB that has a single entry in the TLB
- The physical superpage has to be in contiguous physical memory – the 16 physical pages have to be moved so they are contiguous



Ski Rental Problem

- Promoting a series of contiguous virtual pages into a superpage reduces TLB misses, but has a cost: copying physical memory into contiguous locations

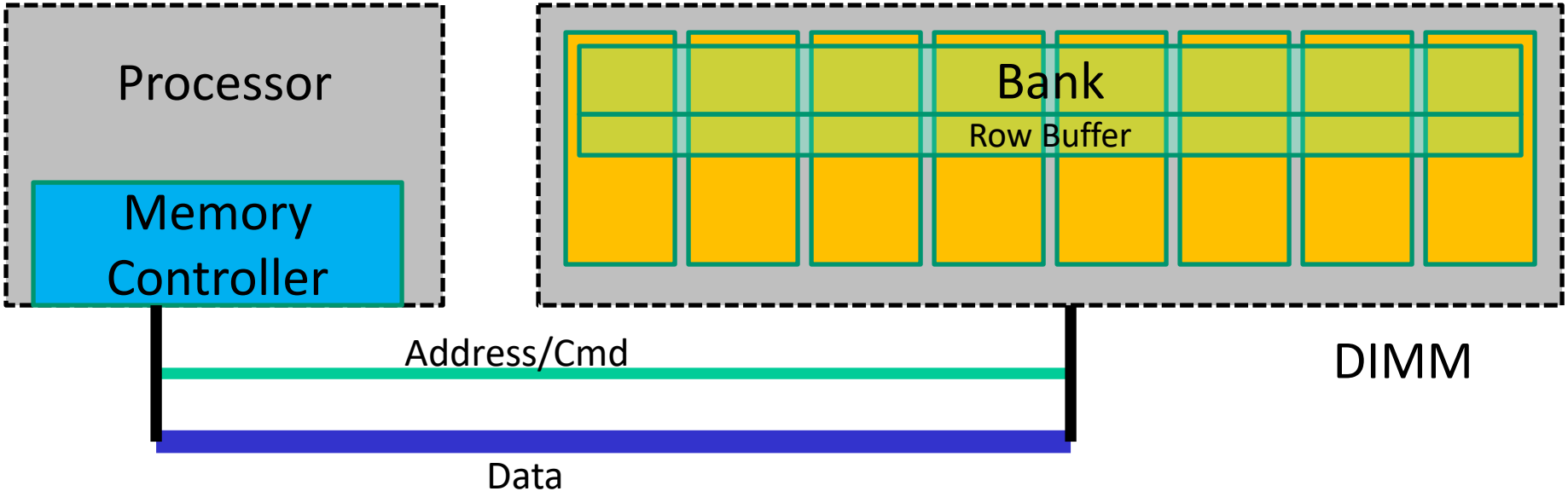
- Page usage statistics can determine if pages are good candidates for superpage promotion, but if cost of a TLB miss is *x* and cost of copying pages is *Nx,* when do you decide to form a superpage?

- If ski rentals cost $50 and new skis cost $500, when do I decide to buy new skis?
  - If I rent 10 times and then buy skis, I'm guaranteed to not spend more than twice the optimal amount

DRAM Main Memory

- Main memory is stored in DRAM cells that have much higher storage density
- DRAM cells lose their state over time – must be refreshed periodically, hence the name *Dynamic*
- DRAM access suffers from long access time and high energy overhead

Memory Architecture



- DIMM: a PCB with DRAM chips on the back and front
- Rank: a collection of DRAM chips that work together to respond to a request and keep the data bus full
- A 64-bit data bus will need 8 x8 DRAM chips or 4 x16 DRAM chips or..
- Bank: a subset of a rank that is busy during one request
- Row buffer: the last row (say, 8 KB) read from a bank, acts like a cache