CENG/CSCI 3420
Computer Organization and Design
Spring 2014

Lecture 08: Exploiting the Memory Hierarchy – B

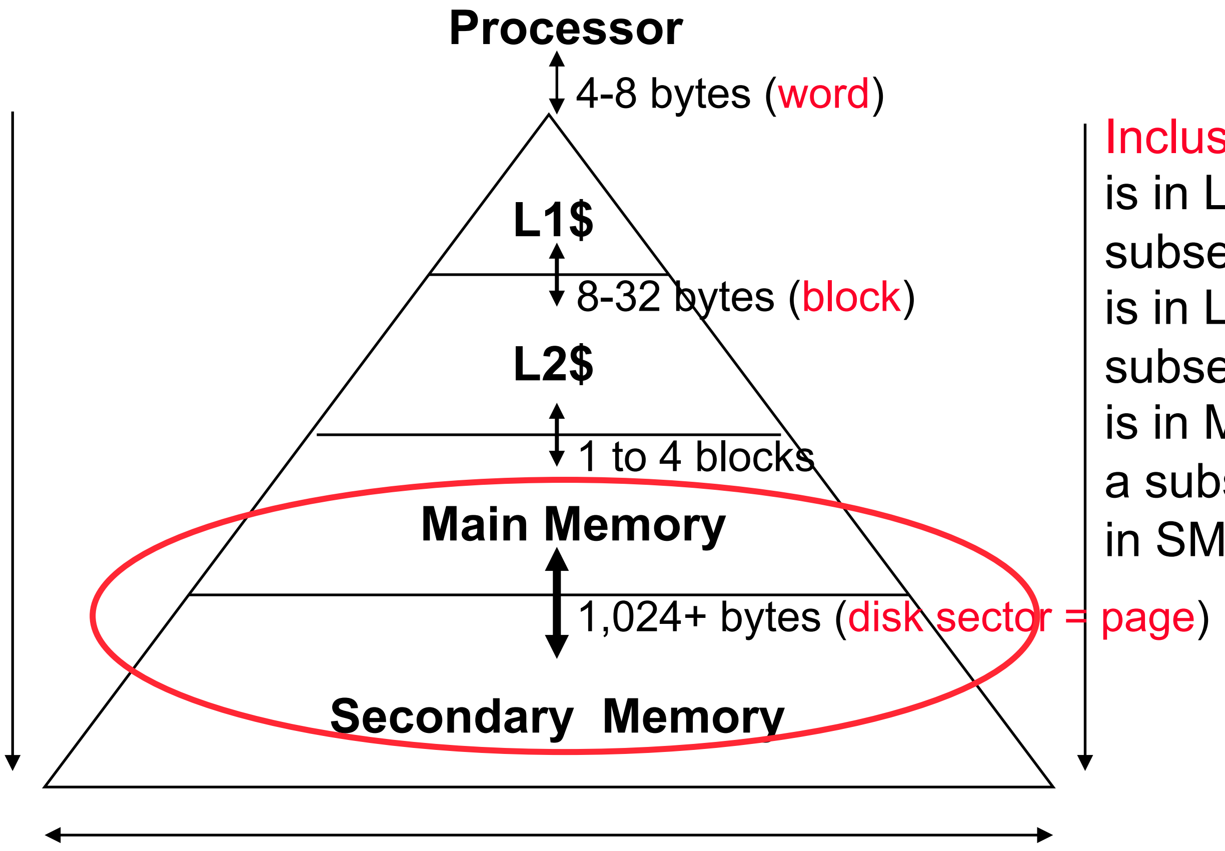
XU, Qiang 徐強

[Adapted from UC Berkeley's D. Patterson's and

from PSU's Mary J. Irwin's slides with additional credits to Y. Xie]

Review: The Memory Hierarchy

- Take advantage of the principle of locality to present the user with as much memory as is available in the cheapest technology at the speed offered by the fastest technology



Increasing distance from the processor in access time

Inclusive– what is in L1$ is a subset of what is in L2$ is a subset of what is in MM that is a subset of is in SM

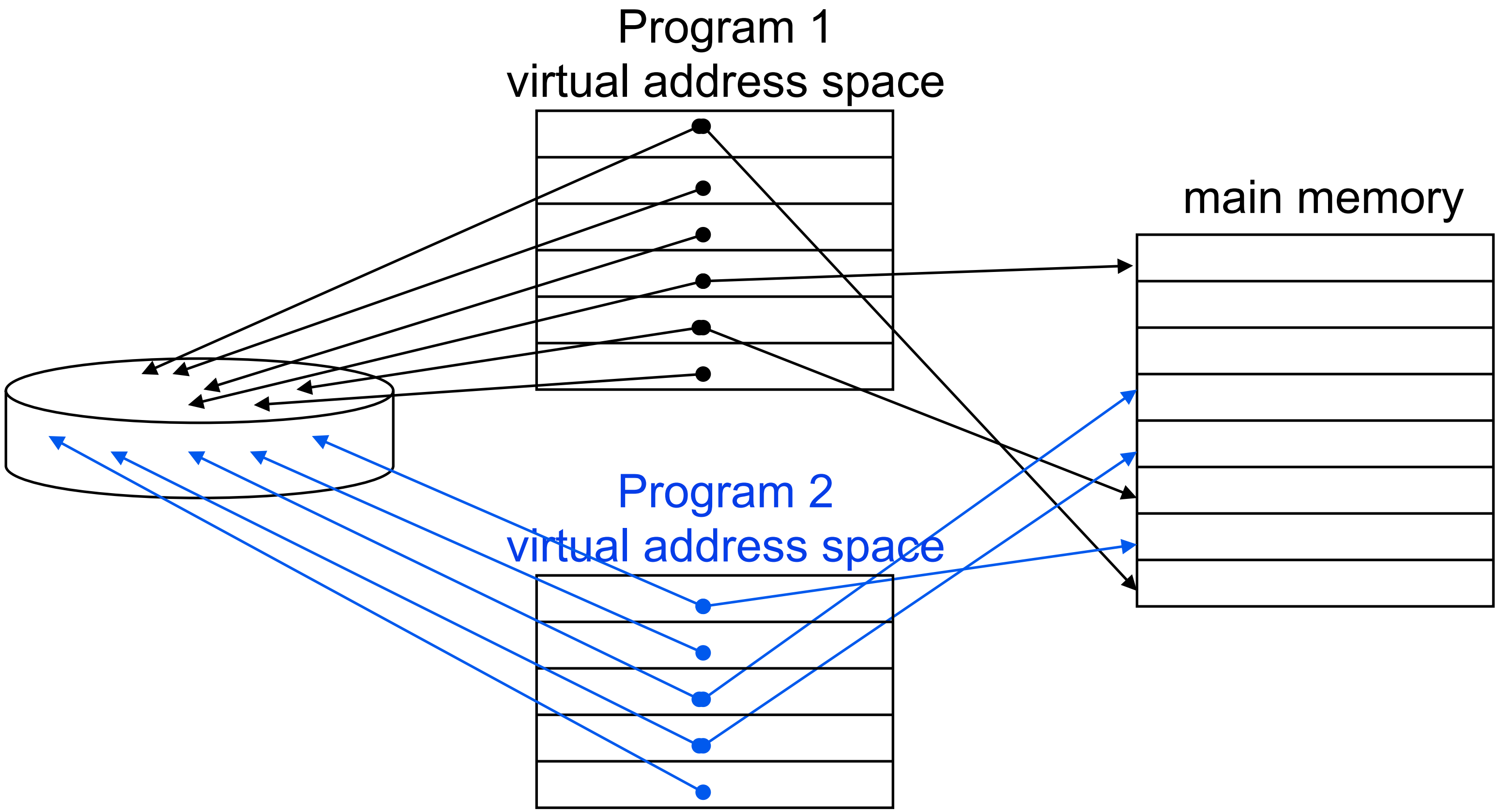
(Relative) size of the memory at each level

Virtual Memory

- Use main memory as a “cache” for secondary memory
  - Allows efficient and safe sharing of memory among multiple programs
  - Provides the ability to easily run programs larger than the size of physical memory
  - Simplifies loading a program for execution by providing code relocation (i.e., the code can be loaded anywhere in main memory)
- What makes it work? – again the Principle of Locality
  - A program is likely to access a relatively small portion of its address space during any period of time
- Each program is compiled into its own address space – a “virtual” address space
  - During run-time each virtual address must be translated to a physical address (an address in main memory)

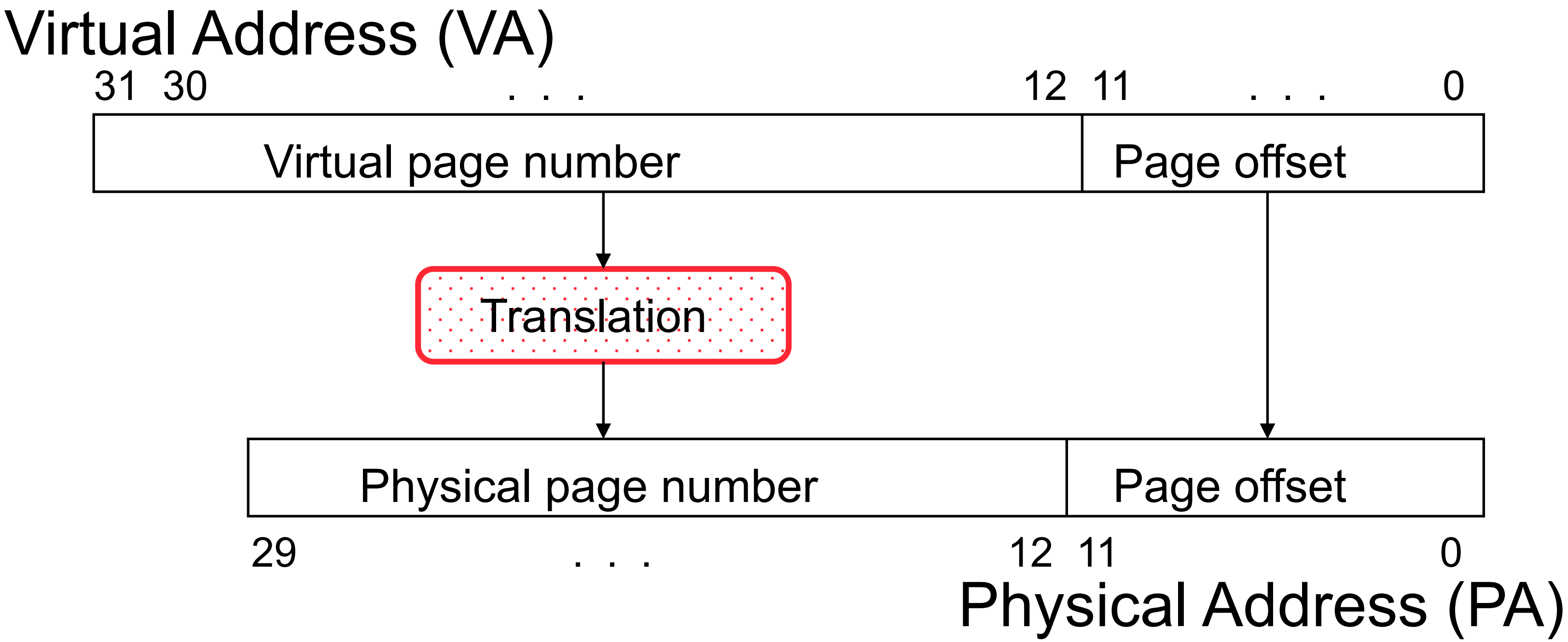
Two Programs Sharing Physical Memory

- A program's address space is divided into pages (all one fixed size) or segments (variable sizes)
  - The starting location of each page (either in main memory or in secondary memory) is contained in the program's page table



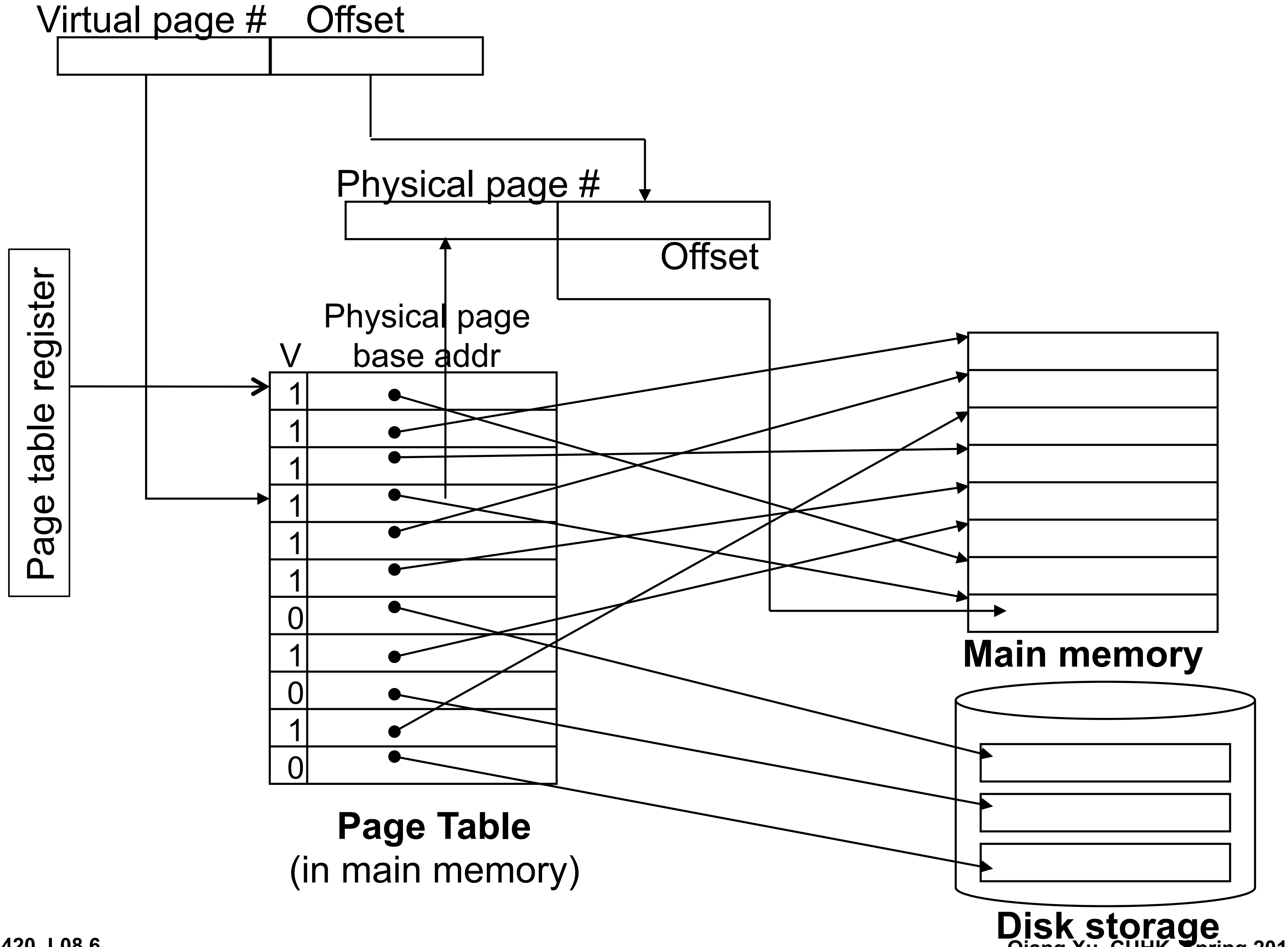
Address Translation

- A virtual address is translated to a physical address by a combination of hardware and software



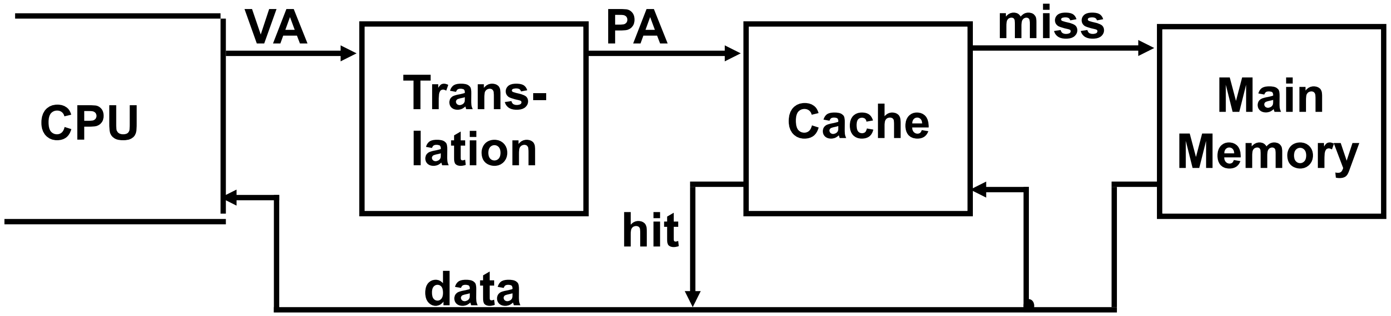
- So each memory request *first* requires an address translation from the virtual space to the physical space
  - A virtual memory miss (i.e., when the page is not in physical memory) is called a page fault

Address Translation Mechanisms



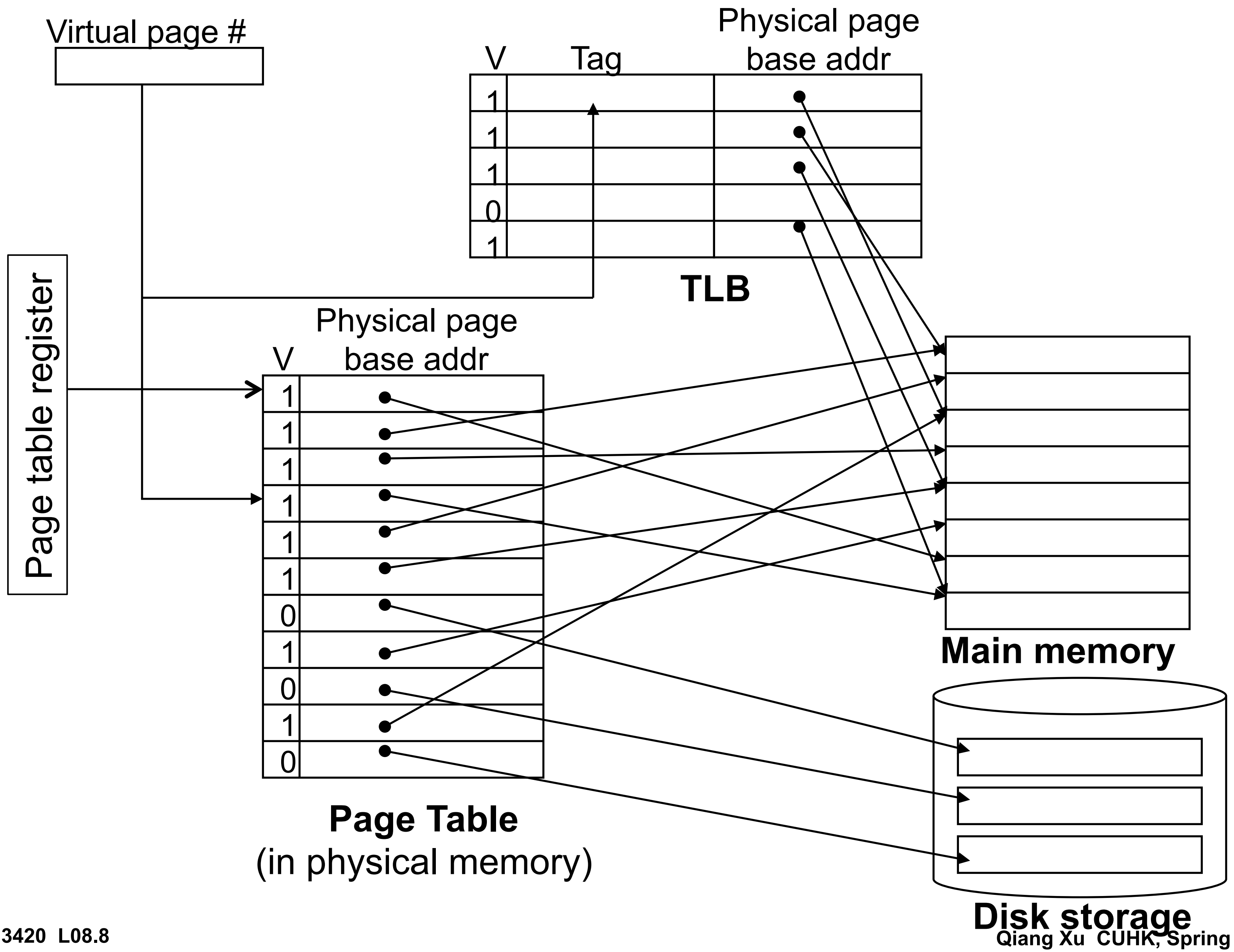
Virtual Addressing with a Cache

- Thus it takes an *extra* memory access to translate a VA to a PA



- This makes memory (cache) accesses very expensive (if every access was really *two* accesses)
- The hardware fix is to use a Translation Lookaside Buffer (TLB) – a small cache that keeps track of recently used address mappings to avoid having to do a page table lookup

Making Address Translation Fast



Page Table
(in physical memory)

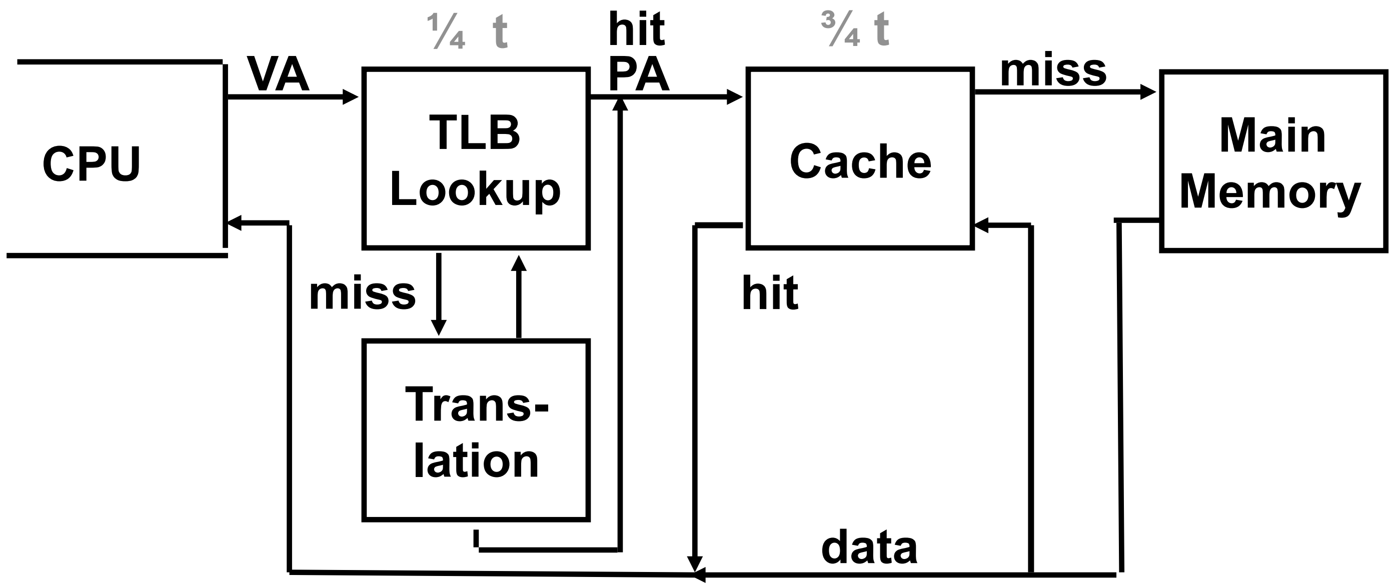
Translation Lookaside Buffers (TLBs)

- Just like any other cache, the TLB can be organized as fully associative, set associative, or direct mapped

V	Virtual Page #	Physical Page #	Dirty	Ref	Access

- TLB access time is typically smaller than cache access time (because TLBs are much smaller than caches)
  - TLBs are typically not more than 512 entries even on high end machines

A TLB in the Memory Hierarchy



- ❑ A TLB miss – is it a page fault or merely a TLB miss?
  - ● If the page is loaded into main memory, then the TLB miss can be handled (in hardware or software) by loading the translation information from the page table into the TLB
    - Takes 10's of cycles to find and load the translation info into the TLB
  - ● If the page is not in main memory, then it's a true page fault
    - Takes 1,000,000's of cycles to service a page fault
- ❑ TLB misses are much more frequent than true page faults

TLB Event Combinations

TLB	Page Table	Cache	Possible? Under what circumstances?
Hit	Hit	Hit	
Hit	Hit	Miss	
Miss	Hit	Hit	
Miss	Hit	Miss	
Miss	Miss	Miss	
Hit	Miss	Miss/ Hit	
Miss	Miss	Hit	

Handling a TLB Miss

- Consider a TLB miss for a page that is present in memory (i.e., the Valid bit in the page table is set)
  - A TLB miss (or a page fault exception) must be asserted by the end of the same clock cycle that the memory access occurs so that the next clock cycle will begin exception processing

Register	CP0 Reg #	Description
`EPC`	14	Where to restart after exception
`Cause`	13	Cause of exception
`BadVAddr`	8	Address that caused exception
`Index`	0	Location in TLB to be read/written
`Random`	1	Pseudorandom location in TLB
`EntryLo`	2	Physical page address and flags
`EntryHi`	10	Virtual page address
`Context`	4	Page table address & page number

A MIPS Software TLB Miss Handler

- When a TLB miss occurs, the hardware saves the address that caused the miss in `BadVAddr` and transfers control to 8000 0000$_{hex}$, the location of the TLB miss handler

```
TLBmiss:
 mfc0  $k1, Context   #copy addr of PTE into $k1
 lw    $k1, 0($k1)    #put PTE into $k1
 mtc0  $k1, EntryLo   #put PTE into EntryLo
 tlbwr                #put EntryLo into TLB
                      #    at Random
 eret                 #return from exception
```

- `tlbwr` copies from `EntryLo` into the TLB entry selected by the control register `Random`

- A TLB miss takes about a dozen clock cycles to handle

Some Virtual Memory Design Parameters

	Paged VM	TLBs
Total size	16,000 to 250,000 words	16 to 512 entries
Total size (KB)	250,000 to 1,000,000,000	0.25 to 16
Block size (B)	4000 to 64,000	4 to 8
Hit time		0.5 to 1 clock cycle
Miss penalty (clocks)	10,000,000 to 100,000,000	10 to 100
Miss rates	0.00001% to 0.0001%	0.01% to 1%

Two Machines' TLB Parameters

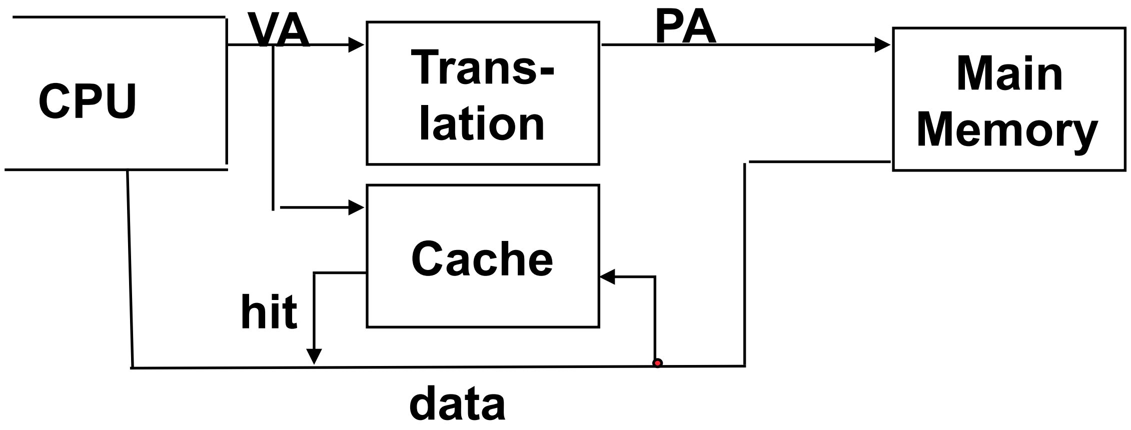
	Intel Nehalem	AMD Barcelona
Address sizes	48 bits (vir); 44 bits (phy)	48 bits (vir); 48 bits (phy)
Page size	4KB	4KB
TLB organization	L1 TLB for instructions and L1 TLB for data per core; both are 4-way set assoc.; LRU L1 ITLB has 128 entries, L2 DTLB has 64 entries L2 TLB (unified) is 4-way set assoc.; LRU L2 TLB has 512 entries TLB misses handled in hardware	L1 TLB for instructions and L1 TLB for data per core; both are fully assoc.; LRU L1 ITLB and DTLB each have 48 entries L2 TLB for instructions and L2 TLB for data per core; each are 4-way set assoc.; round robin LRU Both L2 TLBs have 512 entries TLB misses handled in hardware

Two Machines' TLB Parameters

	Intel P4	AMD Opteron
TLB organization	1 TLB for instructions and 1TLB for data Both 4-way set associative Both use ~LRU replacement Both have 128 entries TLB misses handled in hardware	2 TLBs for instructions and 2 TLBs for data Both L1 TLBs fully associative with ~LRU replacement Both L2 TLBs are 4-way set associative with round-robin LRU Both L1 TLBs have 40 entries Both L2 TLBs have 512 entries TBL misses handled in hardware

Why Not a Virtually Addressed Cache?

- A virtually addressed cache would only require address translation on cache misses

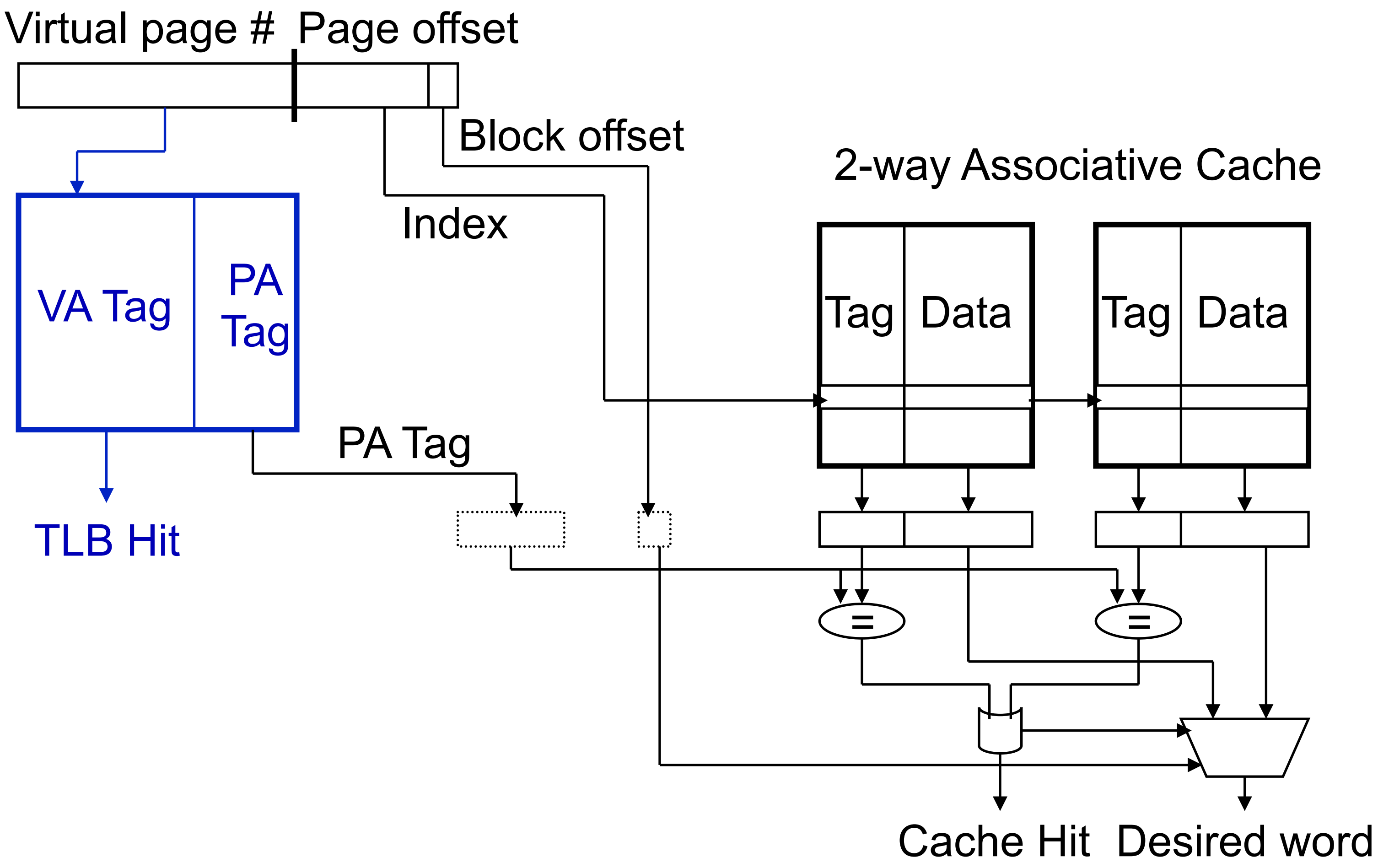


but

- Two programs which are sharing data will have two different virtual addresses for the same physical address – aliasing – so have two copies of the shared data in the cache and two entries in the TBL which would lead to coherence issues
  - Must update all cache entries with the same physical address or the memory becomes inconsistent

Reducing Translation Time

- Can overlap the cache access with the TLB access
  - Works when the high order bits of the VA are used to access the TLB while the low order bits are used as index into cache



The Hardware/Software Boundary

- What parts of the virtual to physical address translation is done by or assisted by the hardware?
  - Translation Lookaside Buffer (TLB) that caches the recent translations
    - TLB access time is part of the cache hit time
    - May allot an extra stage in the pipeline for TLB access
  - Page table storage, fault detection and updating
    - Page faults result in interrupts (precise) that are then handled by the OS
    - Hardware must support (i.e., update appropriately) Dirty and Reference bits (e.g., ~LRU) in the Page Tables
  - Disk placement
    - Bootstrap (e.g., out of disk sector 0) so the system can service a limited number of page faults before the OS is even loaded

4 Questions for the Memory Hierarchy

- ❑ Q1: Where can a entry be placed in the upper level? *(Entry placement)*

- ❑ Q2: How is a entry found if it is in the upper level? *(Entry identification)*

- ❑ Q3: Which entry should be replaced on a miss? *(Entry replacement)*

- ❑ Q4: What happens on a write? *(Write strategy)*

Q1&Q2: Where can a entry be placed/found?

	# of sets	Entries per set
Direct mapped	# of entries	1
Set associative	(# of entries)/ associativity	Associativity (typically 2 to 16)
Fully associative	1	# of entries

	Location method	# of comparisons
Direct mapped	Index	1
Set associative	Index the set; compare set's tags	Degree of associativity
Fully associative	Compare all entries' tags Separate lookup (page) table	# of entries 0

Q3: Which entry should be replaced on a miss?

- Easy for direct mapped – only one choice
- Set associative or fully associative
  - Random
  - LRU (Least Recently Used)

- For a 2-way set associative, random replacement has a miss rate about 1.1 times higher than LRU
- LRU is too costly to implement for high levels of associativity (> 4-way) since tracking the usage information is costly

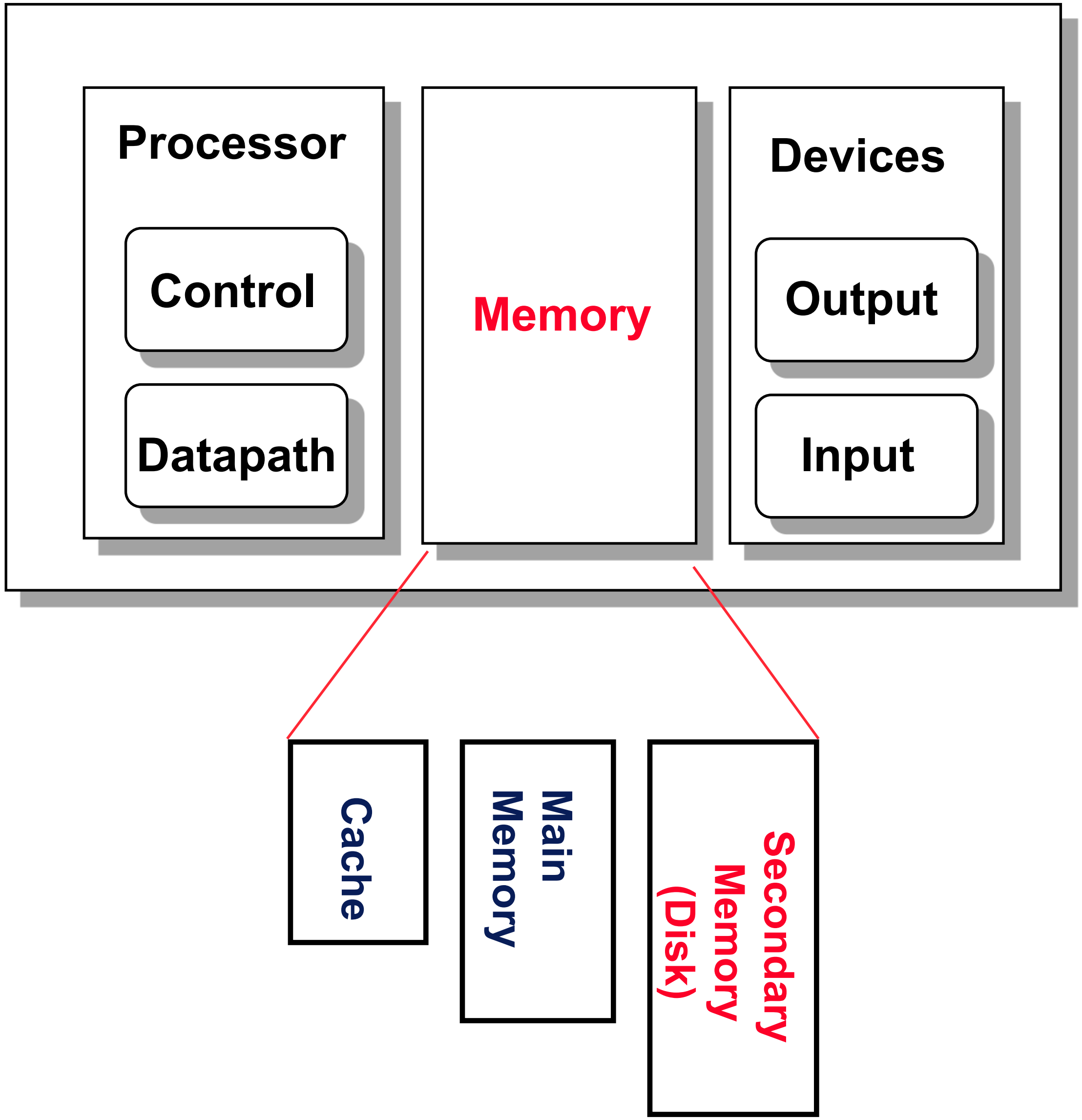
Q4: What happens on a write?

- *Write-through* – The information is written to the entry in the current memory level *and* to the entry in the next level of the memory hierarchy
  - Always combined with a write buffer so write waits to next level memory can be eliminated (as long as the write buffer doesn't fill)
- *Write-back* – The information is written only to the entry in the current memory level. The modified entry is written to next level of memory only when it is replaced.
  - Need a dirty bit to keep track of whether the entry is clean or dirty
  - Virtual memory systems always use write-back of dirty pages to disk
- Pros and cons of each?
  - Write-through: read misses don't result in writes (so are simpler and cheaper), easier to implement
  - Write-back: writes run at the speed of the cache; repeated writes require only one write to lower level

Summary

- The Principle of Locality:
  - Program likely to access a relatively small portion of the address space at any instant of time.
    - Temporal Locality: Locality in Time
    - Spatial Locality: Locality in Space
- Caches, TLBs, Virtual Memory all understood by examining how they deal with the four questions
  1. Where can entry be placed?
  2. How is entry found?
  3. What entry is replaced on miss?
  4. How are writes handled?
- Page tables map virtual address to physical address
  - TLBs are important for fast translation

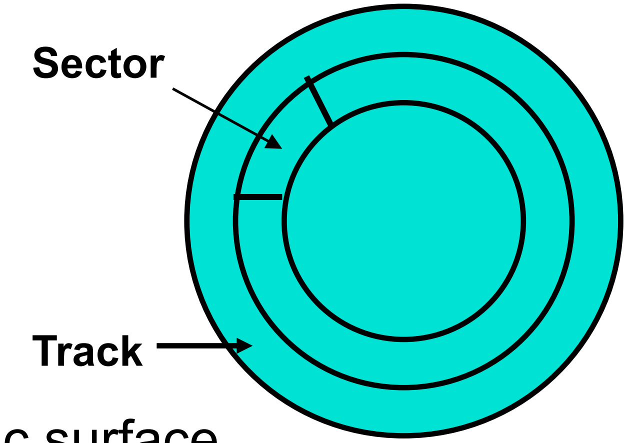
Review: Major Components of a Computer



Magnetic Disk

❑ Purpose

- Long term, nonvolatile storage
- Lowest level in the memory hierarchy
  - slow, large, inexpensive



❑ General structure

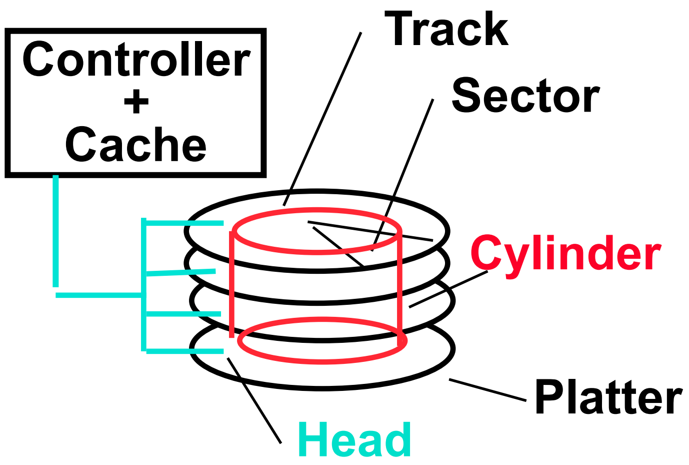
- A rotating platter coated with a magnetic surface
- A moveable read/write head to access the information on the disk

❑ Typical numbers

- 1 to 4 platters (each with 2 recordable surfaces) per disk of 1” to 3.5” in diameter
- Rotational speeds of 5,400 to 15,000 RPM
- 10,000 to 50,000 tracks per surface
  - cylinder - all the tracks under the head at a given point on all surfaces
- 100 to 500 sectors per track
  - the smallest unit that can be read/written (typically 512B)

Magnetic Disk Characteristic

- Disk read/write components



1. Seek time: position the head over the proper track (3 to 13 ms avg)
   - due to locality of disk references the actual average seek time may be only 25% to 33% of the advertised number
2. Rotational latency: wait for the desired sector to rotate under the head (½ of 1/RPM converted to ms)
   - 0.5/5400RPM = 5.6ms to 0.5/15000RPM = 2.0ms
3. Transfer time: transfer a block of bits (one or more sectors) under the head to the disk controller's cache (70 to 125 MB/s are typical disk transfer rates in 2008)
   - the disk controller's "cache" takes advantage of spatial locality in disk accesses
     - cache transfer rates are much faster (e.g., 375 MB/s)
4. Controller time: the overhead the disk controller imposes in performing a disk I/O access (typically < .2 ms)

Typical Disk Access Time

- The average time to read or write a 512B sector for a disk rotating at 15,000 RPM with average seek time of 4 ms, a 100MB/sec transfer rate, and a 0.2 ms controller overhead

If the measured average seek time is 25% of the advertised average seek time, then

- The rotational latency is usually the largest component of the access time

Disk Interface Standards

- Higher-level disk interfaces have a microprocessor disk controller that can lead to performance optimizations
  - ATA (Advanced Technology Attachment) – An interface standard for the connection of storage devices such as hard disks, solid-state drives, and CD-ROM drives. Parallel ATA has been largely replaced by serial ATA.
  - SCSI (Small Computer Systems Interface) – A set of standards (commands, protocols, and electrical and optical interfaces) for physically connecting and transferring data between computers and peripheral devices. Most commonly used for hard disks and tape drives.

- In particular, disk controllers have SRAM disk caches which support fast access to data that was recently read and often also include prefetch algorithms to try to anticipate demand

Magnetic Disk Examples (www.seagate.com)

Feature	Seagate ST31000340NS	Seagate ST973451SS	Seagate ST9160821AS
Disk diameter (inches)	3.5	2.5	2.5
Capacity (GB)	1000	73	160
# of surfaces (heads)	4	2	2
Rotation speed (RPM)	7,200	15,000	5,400
Transfer rate (MB/sec)	105	79-112	44
Minimum seek (ms)	0.8r-1.0w	0.2r-0.4w	1.5r-2.0w
Average seek (ms)	8.5r-9.5w	2.9r-3.3w	12.5r-13.0w
MTTF (hours@25°C)	1,200,000	1,600,000	??
Dim (inches), Weight (lbs)	1x4x5.8, 1.4	0.6x2.8x3.9, 0.5	0.4x2.8x3.9, 0.2
GB/cu.inch, GB/watt	43, 91	11, 9	37, 84
Power: op/idle/sb (watts)	11/8/1	8/5.8/-	1.9/0.6/0.2
Price in 2008, $/GB	~$0.3/GB	~$5/GB	~$0.6/GB

Disk Latency & Bandwidth Milestones

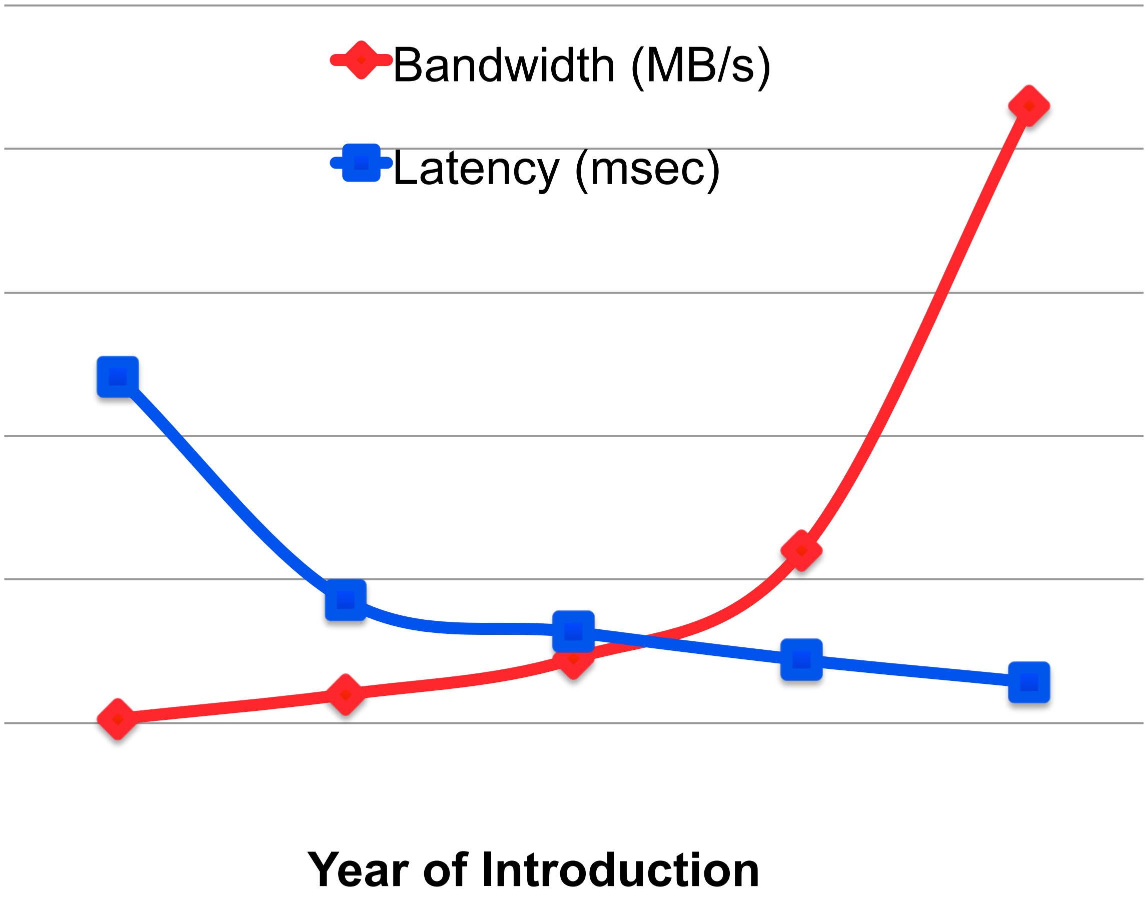
	CDC Wren	SG ST41	SG ST15	SG ST39	SG ST37
RSpeed (RPM)	3600	5400	7200	10000	15000
Year	1983	1990	1994	1998	2003
Capacity (Gbytes)	0.03	1.4	4.3	9.1	73.4
Diameter (inches)	5.25	5.25	3.5	3.0	2.5
Interface	ST-412	SCSI	SCSI	SCSI	SCSI
Bandwidth (MB/s)	0.6	4	9	24	86
Latency (msec)	48.3	17.1	12.7	8.8	5.7

Patterson, CACM Vol 47, #10, 2004

- Disk latency is one average seek time plus the rotational latency.
- Disk bandwidth is the peak transfer time of formatted data from the media (not from the cache).

Latency & Bandwidth Improvements

- In the time that the disk bandwidth doubles the latency improves by a factor of only 1.2 to 1.4



Flash Storage

- Flash memory is the first credible challenger to disks. It is semiconductor memory that is nonvolatile like disks, but has latency 100 to 1000 times faster than disk and is smaller, more power efficient, and more shock resistant.
  - In 2008, the price of flash is $4 to $10 per GB or about 2 to 10 times higher than disk and 5 to 10 times lower than DRAM.
  - Flash memory bits wear out (unlike disks and DRAMs), but wear leveling can make it unlikely that the write limits of the flash will be exceeded

Feature	Kingston	Transend	RiDATA
Capacity (GB)	8	16	32
Bytes/sector	512	512	512
Transfer rates (MB/sec)	4	20r-18w	68r-50w
MTTF	>1,000,000	>1,000,000	>4,000,000
Price (2008)	~ $30	~ $70	~ $300

Dependability, Reliability, Availability

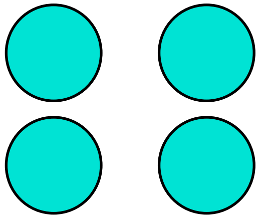
- Reliability – measured by the mean time to failure (MTTF). Service interruption is measured by mean time to repair (MTTR)
- Availability – a measure of service accomplishment

  Availability = MTTF/(MTTF + MTTR)

- To increase MTTF, either improve the quality of the components or design the system to continue operating in the presence of faulty components
  1. Fault avoidance: preventing fault occurrence by construction
  2. Fault tolerance: using redundancy to correct or bypass faulty components (hardware)
     - Fault detection versus fault correction
     - Permanent faults versus transient faults

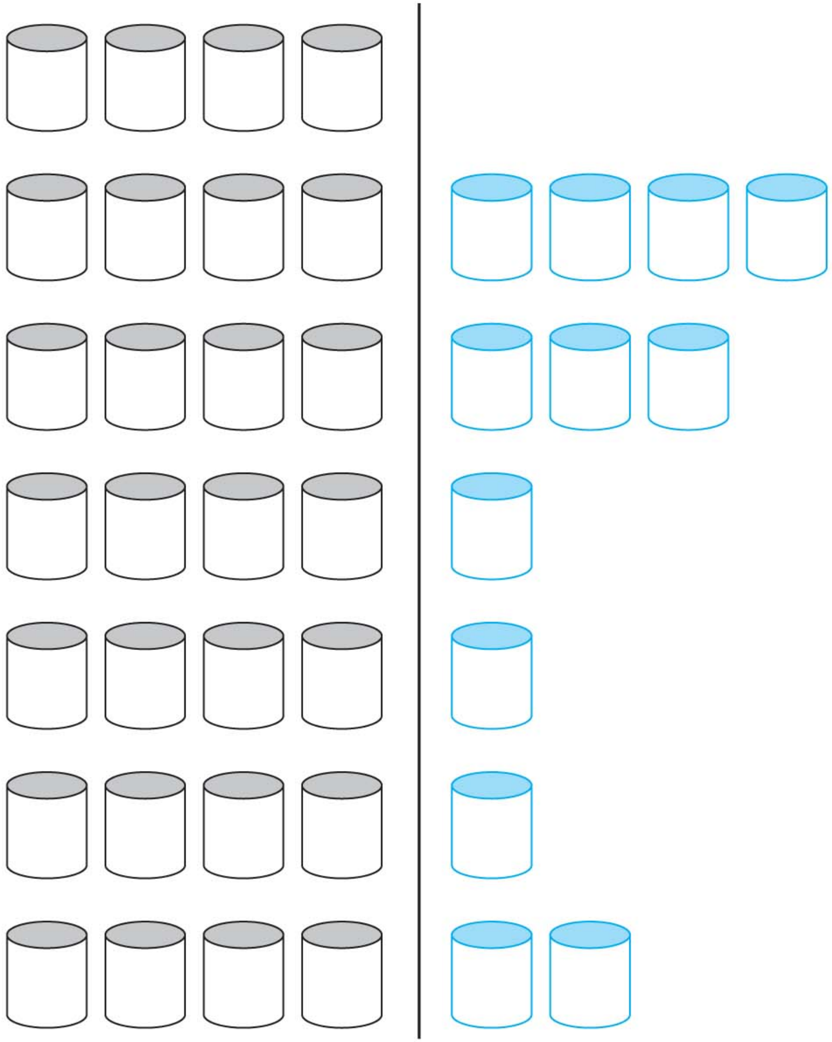
RAIDs: Disk Arrays

Redundant Array of
Inexpensive Disks



- Arrays of small and inexpensive disks
  - Increase potential throughput by having many disk drives
    - Data is spread over multiple disk
    - Multiple accesses are made to several disks at a time
- Reliability is lower than a single disk
- But availability can be improved by adding redundant disks (RAID)
  - Lost information can be reconstructed from redundant information
  - MTTR: mean time to repair is in the order of hours
  - MTTF: mean time to failure of disks is tens of years

RAID Configurations



Data disks

Redundant check disks

RAID 0
(No redundancy)
Widely used

RAID 1
(Mirroring)
EMC, HP(Tandem), IBM

RAID 2
(Error detection and
correction code) Unused

RAID 3
(Bit-interleaved parity)
Storage concepts

RAID 4
(Block-interleaving parity)
Network appliance

RAID 5
(Distributed block-
interleaved parity)
Widely used

RAID 6
(P + Q redundancy)
Recently popular