

CPS 104
Computer Organization and Programming
Lecture 21: Virtual Memory, I/O

Nov. 12, 1999

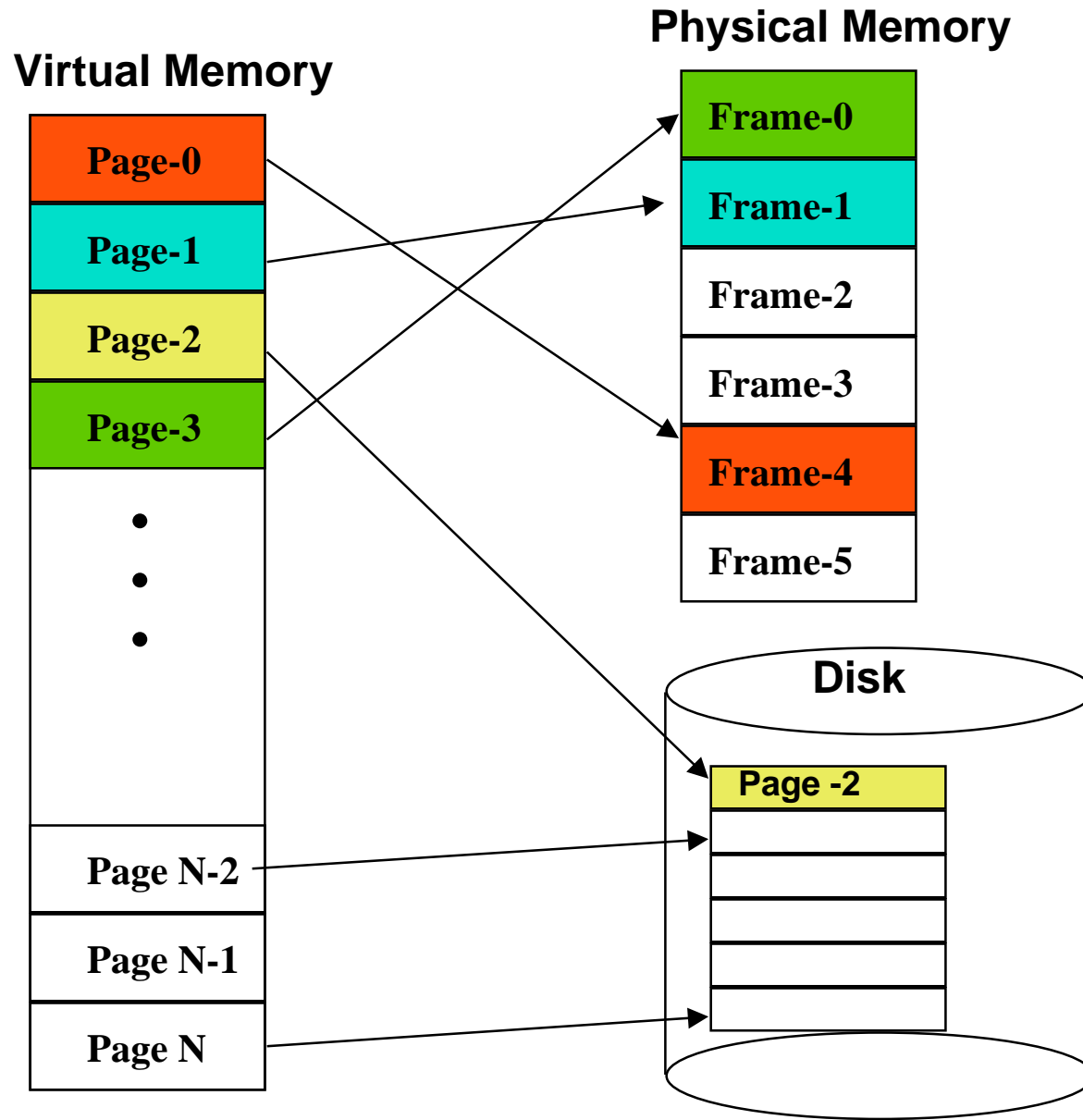
Dietolf (Dee) Ramm

<http://www.cs.duke.edu/~dr/cps104.html>

Outline of Today's Lecture

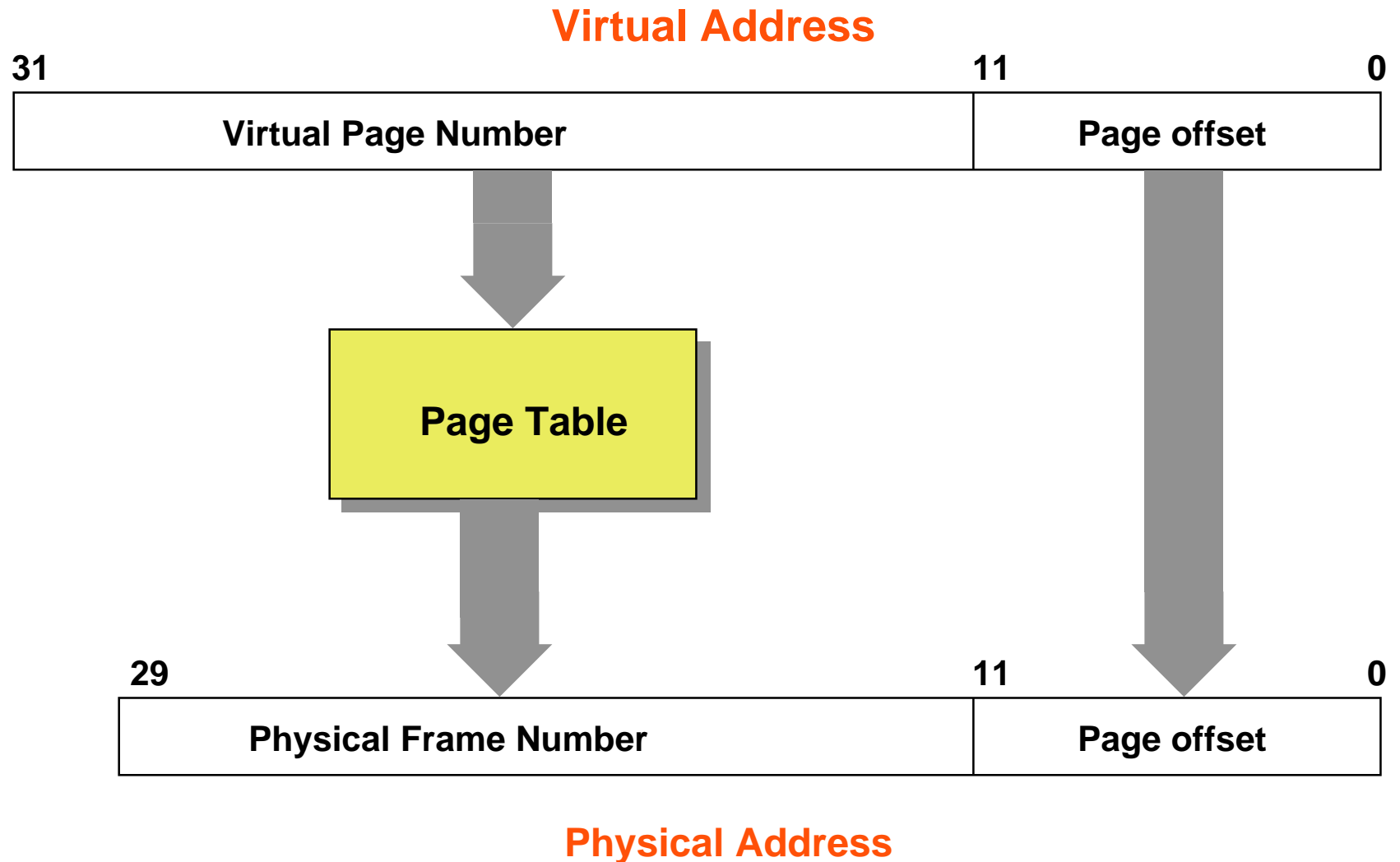
- Review Virtual Memory.
 - * Paged virtual memory.
 - * Virtual to Physical translation: The page table.
 - * Page replacement policies.
- Reducing the **Virtual** to **Physical** address translation time.
 - * The **TLB**
 - * Parallel access to the TLB and Cache
- Memory Protection
- Putting it all together: The SPARC-20 memory system
- I/O: the big picture
- Storage Devices: Disk and tape.

Virtual and Physical Memories



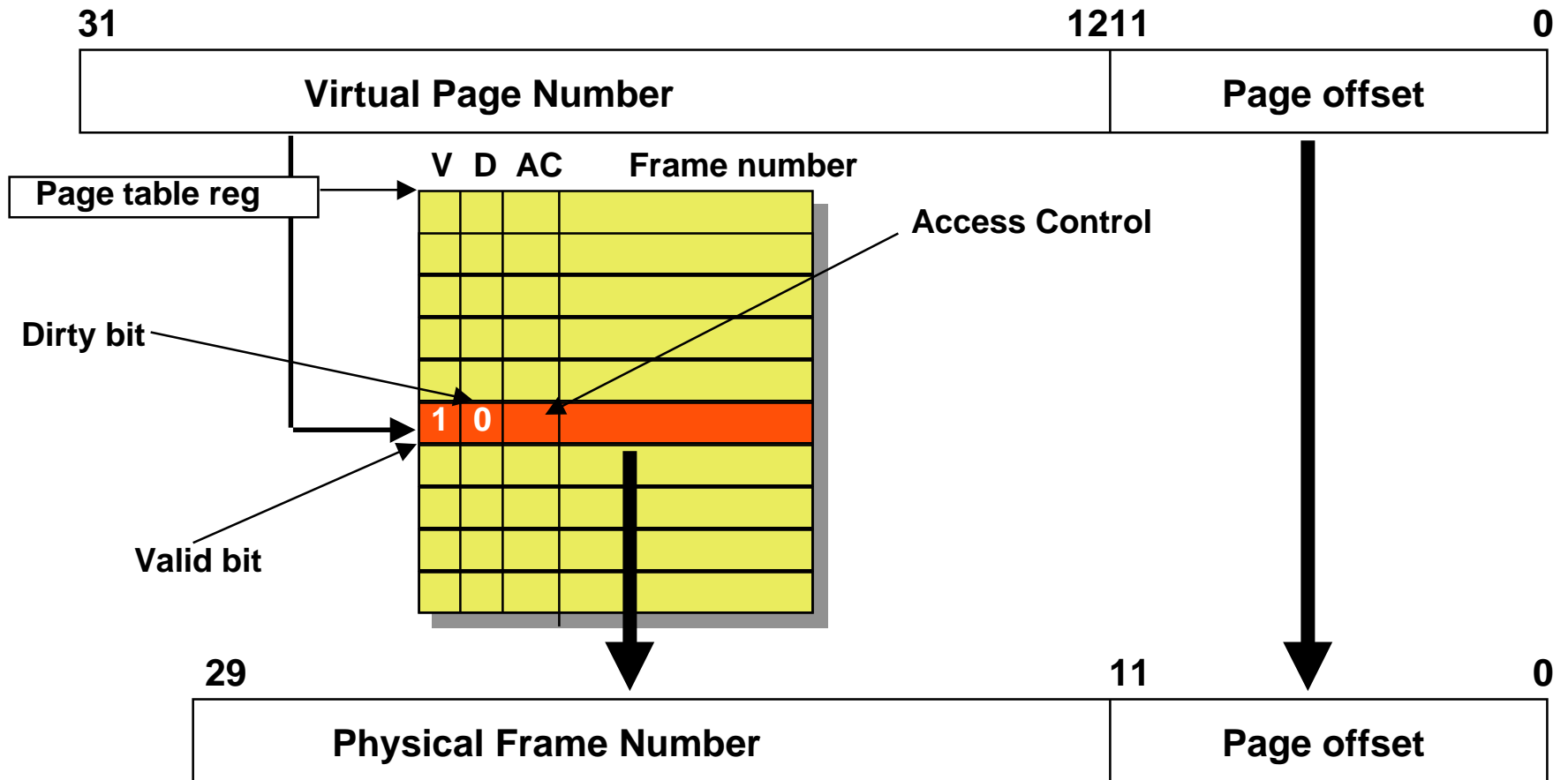
Virtual to Physical Address translation

Page size: 4K



The page table

Virtual Address



Physical Address

Page Replacement Algorithms

Just like cache block replacement!

Least Recently Used:

- selects the least recently used page for replacement
- requires knowledge about past references, more difficult to implement (thread through page table entries from most recently referenced to least recently referenced; when a page is referenced it is placed at the head of the list; the end of the list is the page to replace)
- good performance, recognizes principle of locality

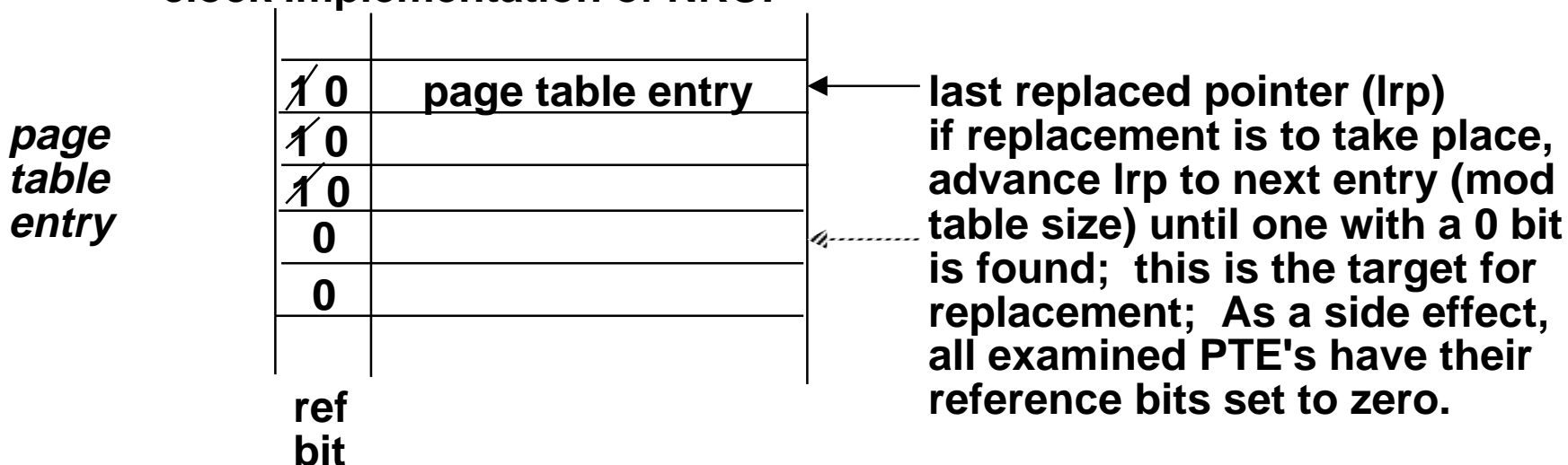
Page Replacement (Continued)

Not Recently Used:

Associated with each page is a reference flag such that
ref flag = 1 if the page has been referenced in recent past
= 0 otherwise

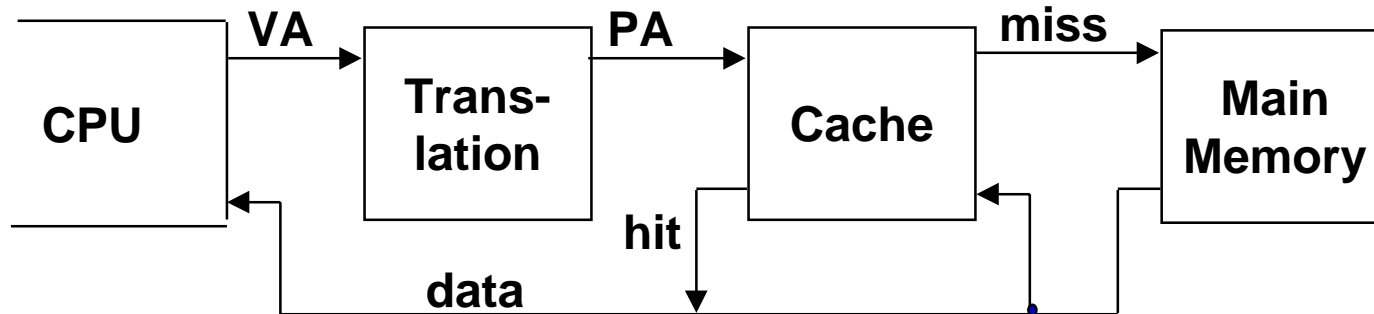
-- if replacement is necessary, choose any page frame such that its reference bit is 0. This is a page that has not been referenced in the recent past

-- clock implementation of NRU:



An optimization is to search for a page that is both
not recently referenced AND not dirty. Why?

Virtual Address and a Cache



It takes an extra memory access to translate VA to PA

This makes cache access very expensive, and this is the "innermost loop" that you want to go as fast as possible

Translation Lookaside Buffer (TLB)

A way to speed up translation is to use a **special cache of recently used page table entries** -- this has many names, but the most frequently used is *Translation Lookaside Buffer* or **TLB**

Virtual Address	Physical Address	Dirty	Ref	Valid	Access

**TLB access time comparable to cache access time
(much less than main memory access time)**

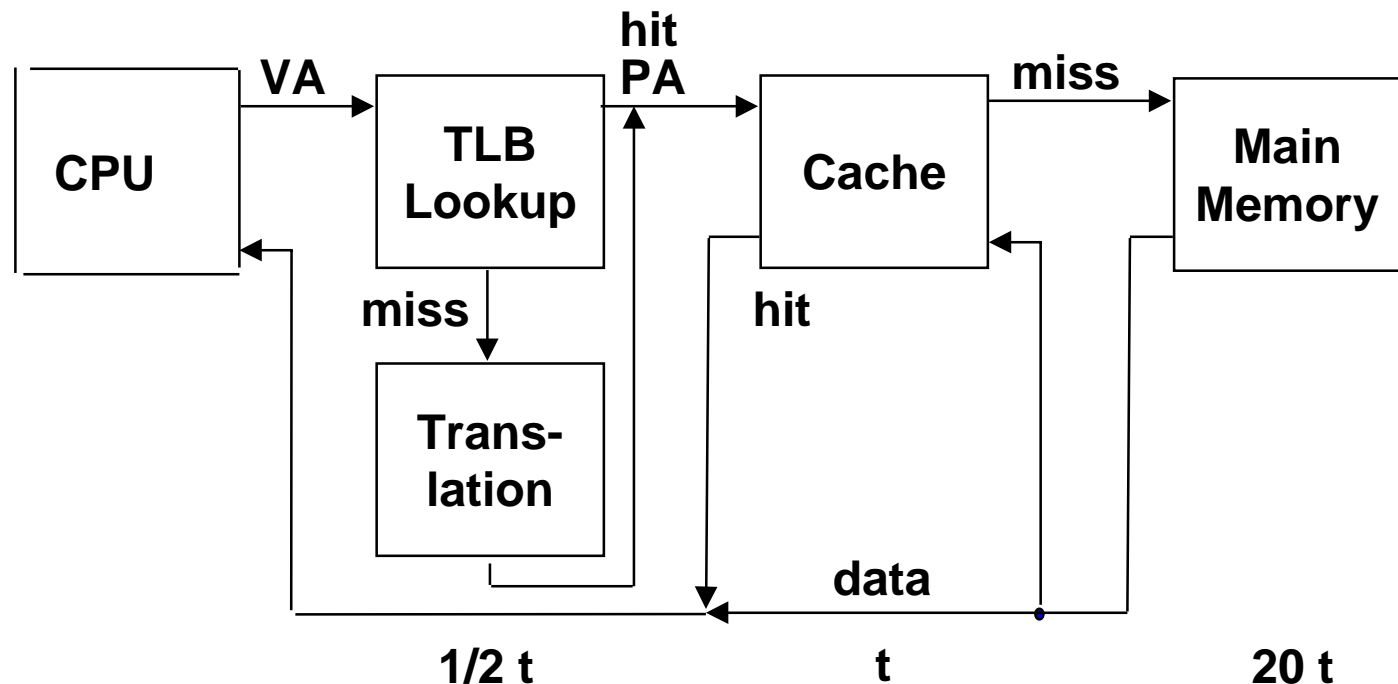
Typical TLB is 64-256 entries fully associative cache with random replacement

Translation Look-Aside Buffers

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

TLBs are usually small, typically not more than 128 - 256 entries even on high end machines. This permits fully associative lookup on these machines. Many mid-range machines use small n-way set associative organizations.

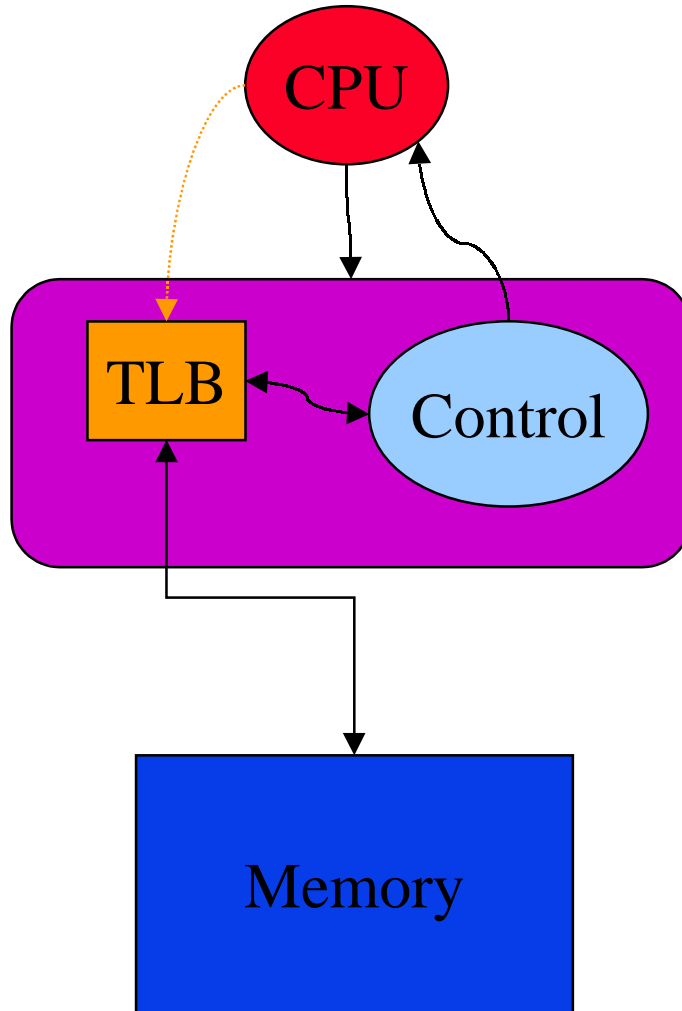
Translation with a TLB



TLB Design

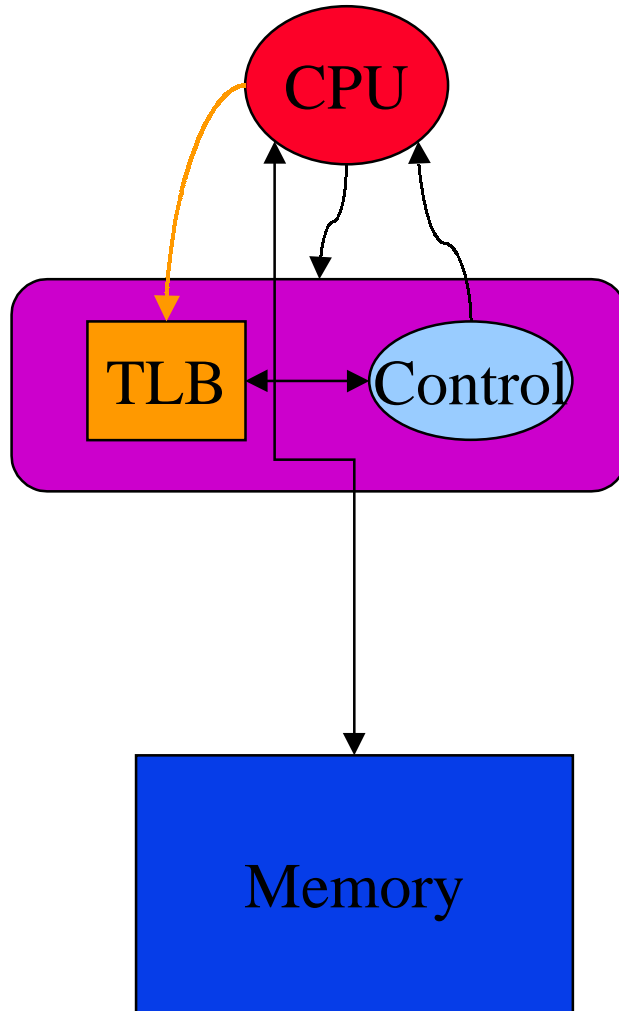
- **Must be fast, not increase critical path**
- **Must achieve high hit ratio**
- **Generally small highly associative (64-128 entries FA cache)**
- **Mapping change**
 - * page added/removed from physical memory
 - * processor must invalidate the TLB entry (**special instructions**)
- **PTE is per process entity**
 - * Multiple processes with same virtual addresses
 - * Context Switches?
- **Flush TLB**
- **Add ASID (PID)**
 - * part of processor state, must be set on context switch

Hardware Managed TLBs



- Hardware Handles TLB miss
- Dictates page table organization
- Complicated state machine to “walk page table”
- Exception only if access violation

Software Managed TLBs



- **Software Handles TLB miss**
 - * OS reads translations from Page Table and puts them in TLB
 - * special instructions
- **Flexible page table organization**
- **Simple Hardware to detect Hit or Miss**
- **Exception if TLB miss or access violation**

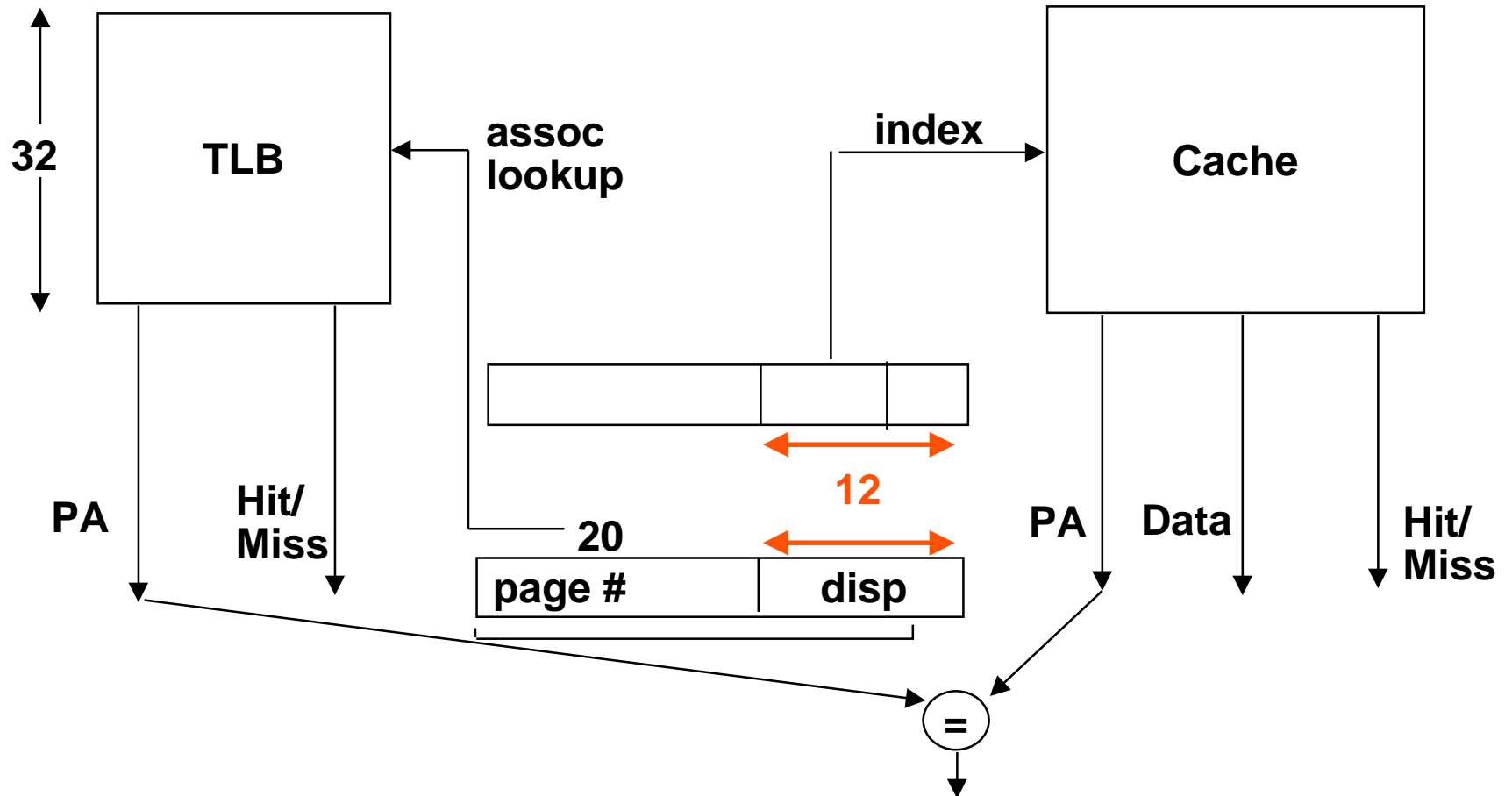
Reducing Translation Time

Machines with TLBs go one step further to reduce # cycles/cache access

They overlap the cache access with the TLB access

**Works because high order bits of the VA are used to look in the TLB
while low order bits are used as index into cache**

Overlapped Cache & TLB Access



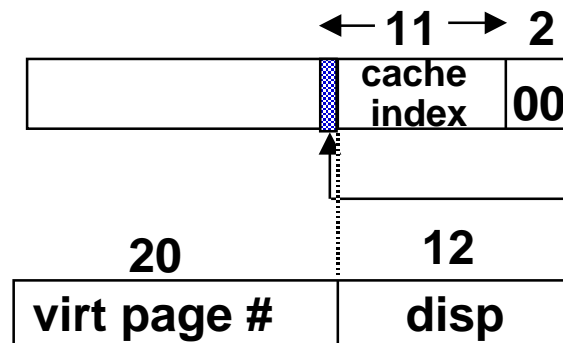
**IF cache hit AND (cache tag = PA) then deliver data to CPU
ELSE IF [cache miss OR (cache tag = PA)] and TLB hit THEN
access memory with the PA from the TLB
ELSE do standard VA translation**

Problems With Overlapped TLB Access

Overlapped access only works as long as the address bits used to index into the cache **do not change** as the result of VA->PA translation

This usually limits things to small caches, large page sizes, or high n-way set associative caches if you want a large cache

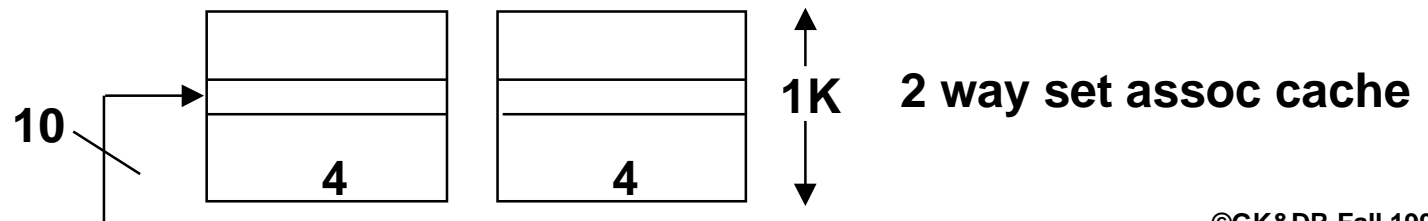
Example: suppose everything the same except that the cache is increased to 8 K bytes instead of 4 K:



This bit is changed by VA translation, but is needed for cache lookup

Solutions:

go to 8K byte page sizes;
go to 2 way set associative cache; or
SW guarantee VA[13]=PA[13]



More on Selecting a Page Size

- Reasons for larger page size

- * Page table size is inversely proportional to the page size.
- * faster cache hit time when cache " page size;
bigger page => bigger cache (no aliasing problem).
- * Transferring larger pages to or from secondary storage, is more efficient (Higher bandwidth)
- * The number of TLB entries is restricted by clock cycle time, so a larger page size reduces TLB misses.

- Reasons for a smaller page size

- * don't waste storage; data must be contiguous within page.
- * quicker process start for small processes(?)

- **Hybrid solution:** multiple page sizes:

Alpha, UltraSPARC: 8KB, 64KB, 512 KB, 4 MB pages

Memory Protection

- **Paging Virtual memory provides protection by:**
 - * **Each process (user or OS) has different virtual memory space.**
 - * **The OS maintain the page tables for all processes.**
 - * **A reference outside the process allocated space cause an exception that lets the OS decide what to do.**
 - * **Memory sharing between processes is done via different Virtual spaces but common physical frames.**

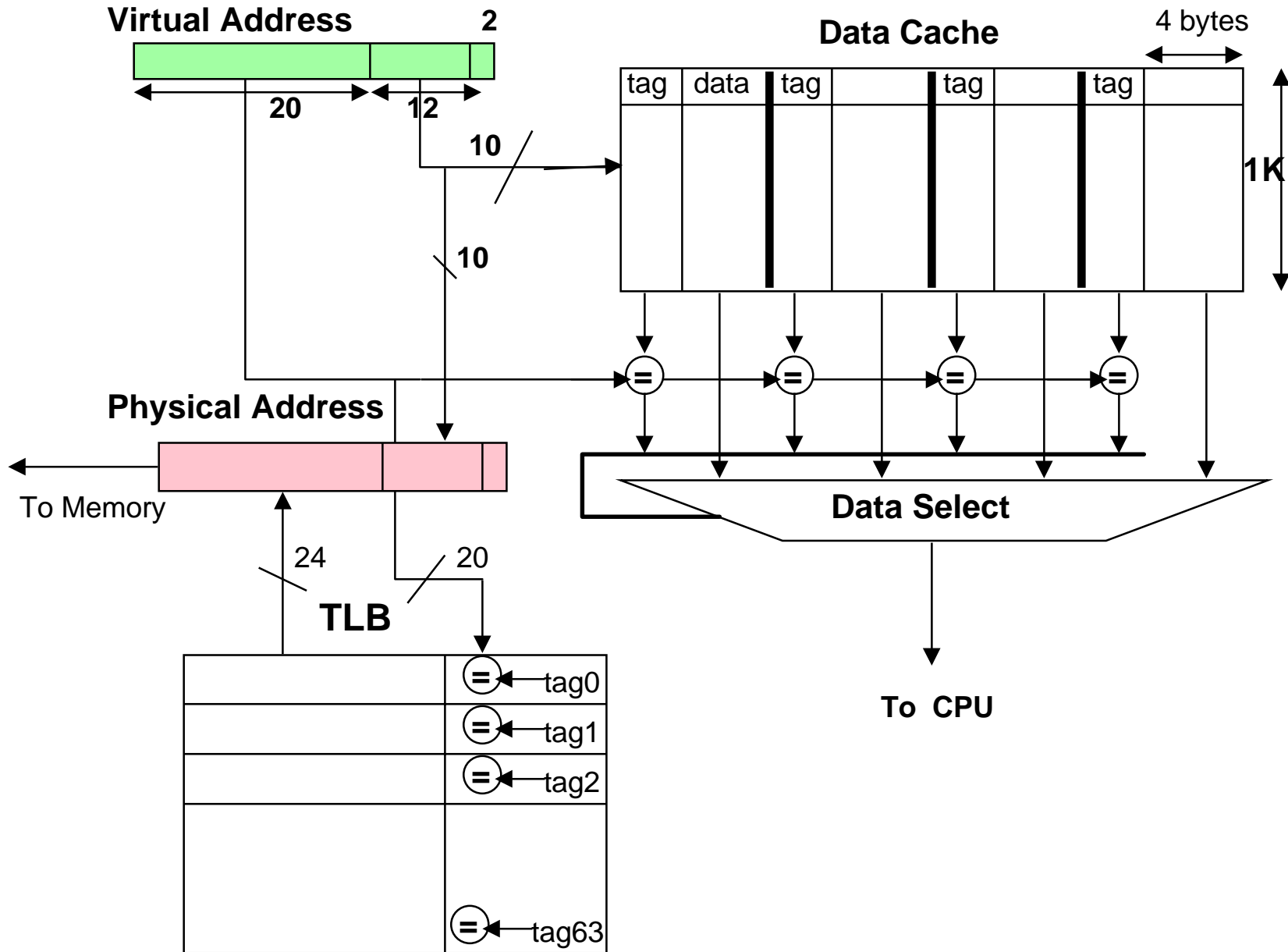
Putting it together: The SparcStation 20:

- The SparcStation 20 has the following memory system.
- Caches: Two level-1 caches: **I-cache** and **D-cache**

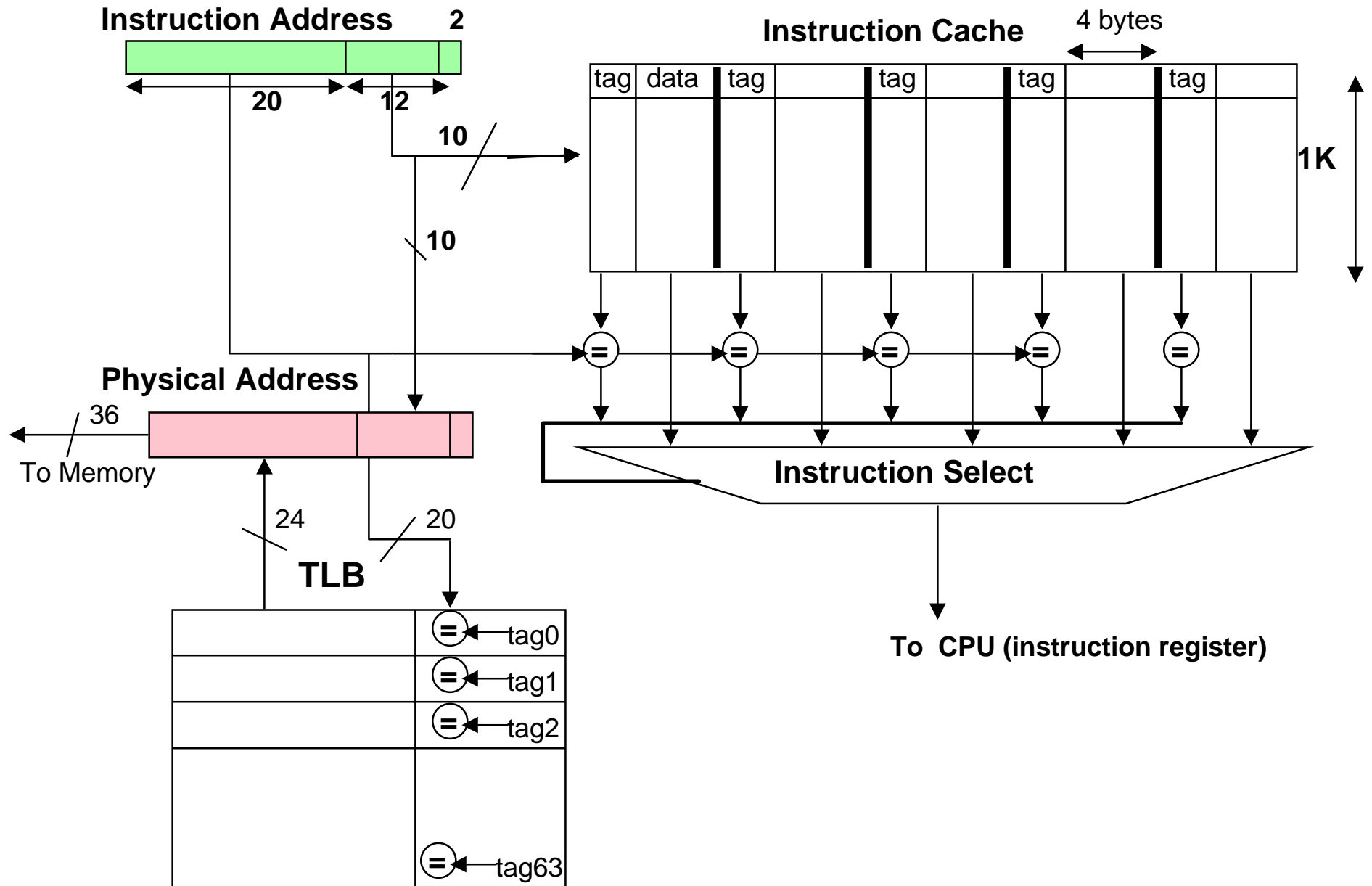
Parameter	Instruction cache	Data cache
Organization	20Kbyte 5-way SA	16KB 4-way SA
Page size	4K bytes	4K bytes
Line size	8 bytes	4 bytes
Replacement	Pseudo LRU	Pseudo LRU

- **TLB:** 64 entry Fully Associative TLB, Random replacement
- **External Level-2 Cache:** 1M-byte, Direct Map, 128 byte blocks, 32-byte sub-blocks.

SparcStation 20 Data Access



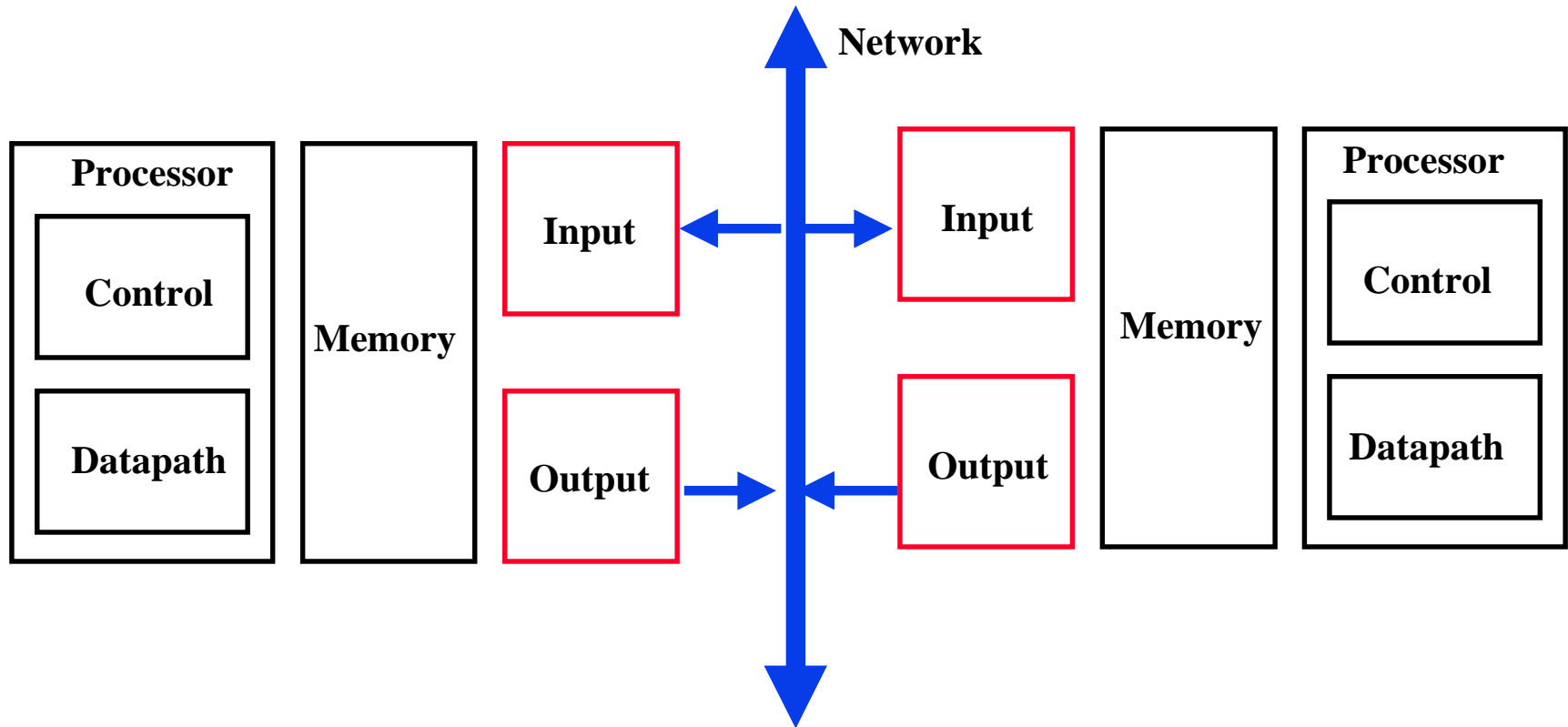
SparcStation 20: Instruction Memory



Input / Output

The Big Picture: Where are We Now?

- Today's Topic: I/O Systems

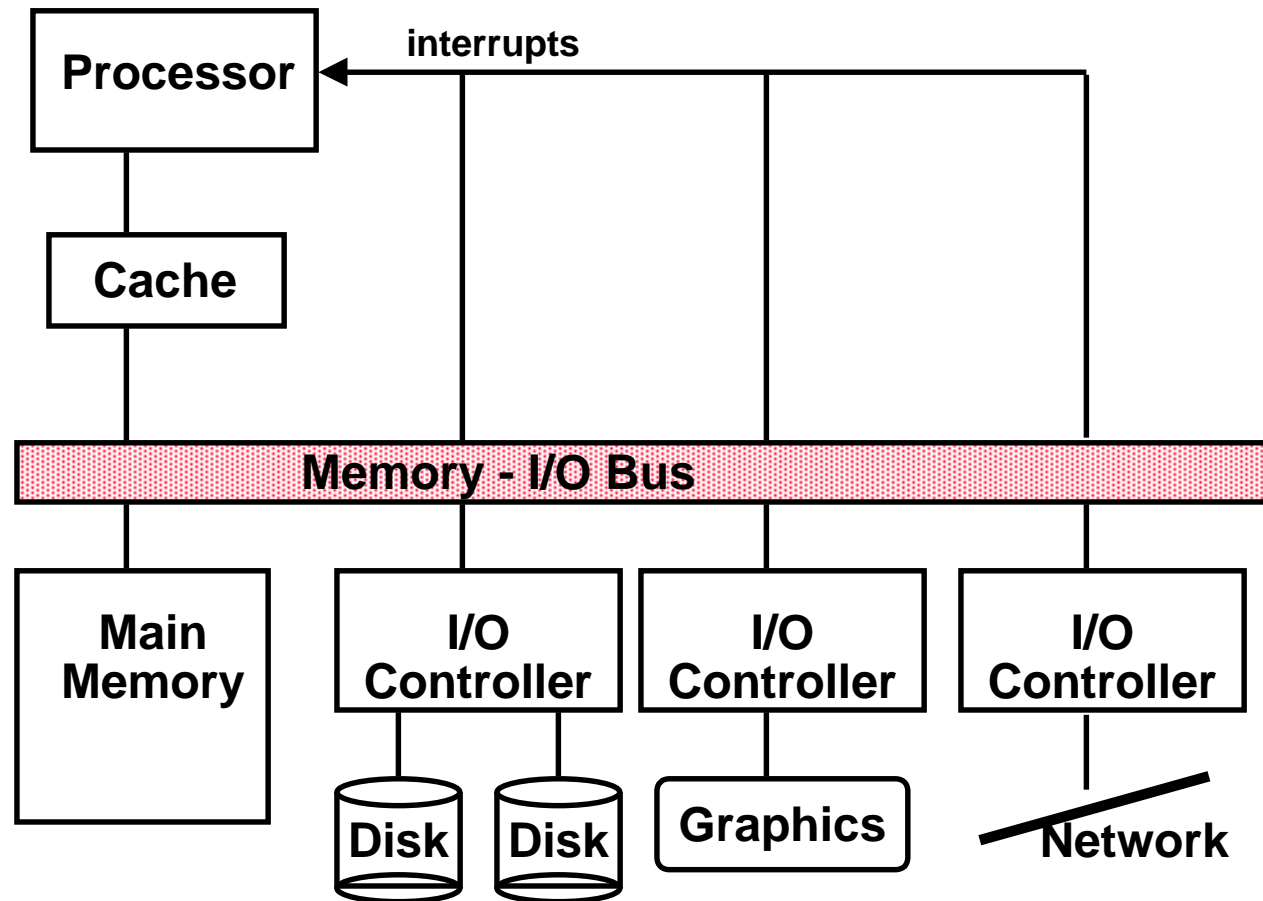


Motivation: Who Cares About I/O?

- CPU Performance goes up : 50% - 80% per year
- I/O system performance limited by *mechanical delays*
< 5% per year (IO per sec or MB per sec)
- Amdahl's Law: system speed-up limited by the slowest part!
10% IO & 10x CPU => 5x Performance (lose 50%)
10% IO & 100x CPU => 10x Performance (lose 90%)
- I/O bottleneck:
Diminishing fraction of time in CPU
Diminishing value of faster CPUs

I/O System Design Issues

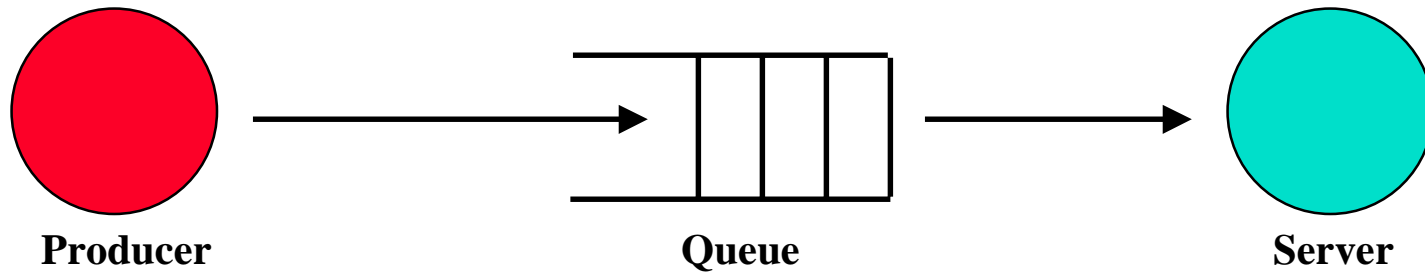
- * Performance
- * Expandability
- * Resilience in the face of failure



I/O System Performance

- **I/O System performance depends on many aspects of the system (“weakest link in the chain”):**
 - * **The CPU**
 - * **The memory system:**
 - **Internal and external caches**
 - **Main Memory**
 - * **The underlying interconnection (buses)**
 - * **The I/O controller**
 - * **The I/O device**
 - * **The speed of the I/O software (Operating System)**
 - * **The efficiency of the software’s use of the I/O devices**
- **Two common performance metrics:**
 - * **Throughput: I/O bandwidth**
 - * **Response time: Latency**

Producer-Server Model



- **Throughput:**

- * The number of tasks completed by the server in unit time
- * In order to get the highest possible throughput:
 - The server should never be idle
 - The queue should never be empty

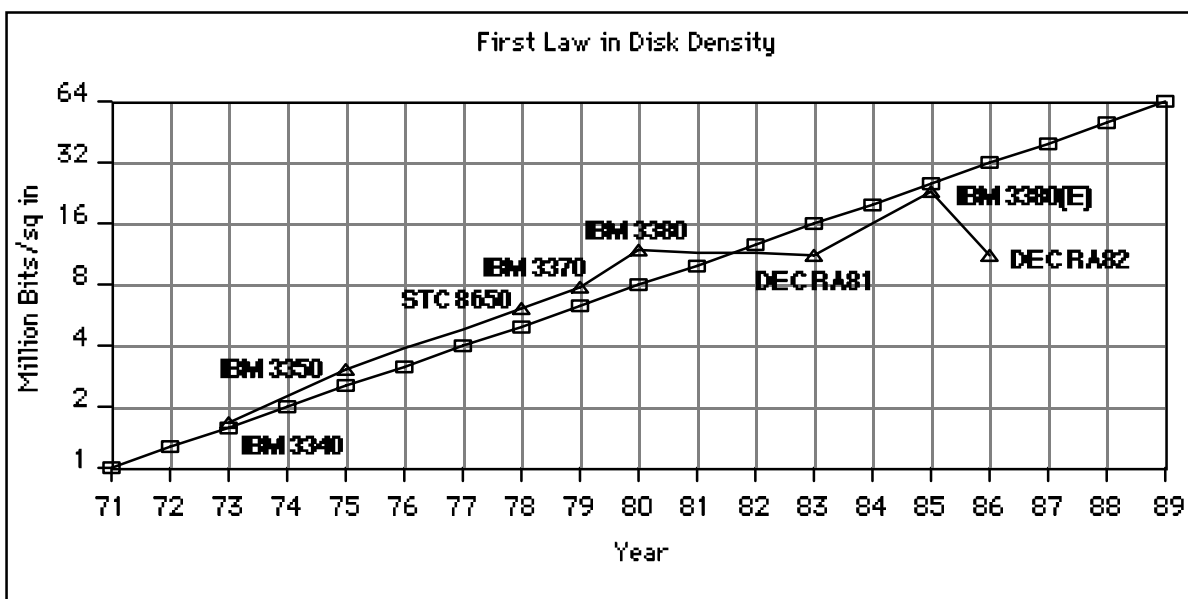
- **Response time:**

- * Begins when a task is placed in the queue
- * Ends when it is completed by the server
- * In order to minimize the response time:
 - The queue should be empty
 - The server will be idle

I/O Device Examples

Device	Behavior	Partner	Data Rate (KB/sec)
Keyboard	Input	Human	0.01
Mouse	Input	Human	0.02
Line Printer	Output	Human	1.00
Laser Printer	Output	Human	100.00
Graphics Displa	Output	Human	30,000.00
Network-LAN	Input/Output	Machine	10,000.00
Floppy disk	Storage	Machine	50.00
Optical Disk	Storage	Machine	500.00
Magnetic Disk	Storage	Machine	5,000.00

Technology Trends



**Disk Capacity
doubles every
1.5 years**

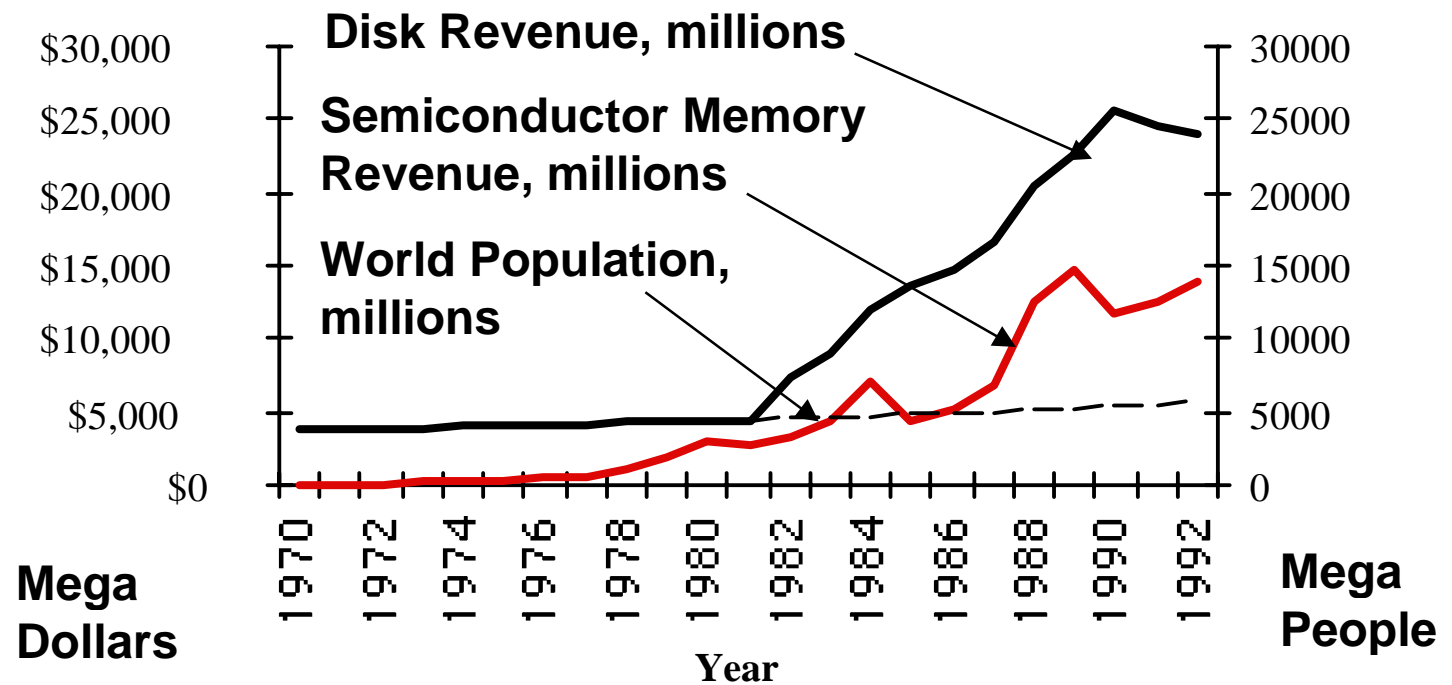
- Today: Processing Power Doubles Every 18 months
- Today: Memory Size Doubles Every 18 months(?)
- Today: Disk Capacity Doubles Every 12-18 months
- *Disk Positioning Rate (Seek + Rotate) Doubles Every Ten Years!*

**The I/O
GAP**

Storage Technology Drivers

- **Driven by the prevailing computing paradigm**
 - 1950s: migration from batch to on-line processing
 - 1990s: migration to ubiquitous computing
 - » computers in phones, books, cars, video cameras, ...
 - » nationwide fiber optical network with wireless tails
- **Effects on storage industry:**
 - Embedded storage
 - » smaller, cheaper, more reliable, lower power
 - Data utilities
 - » high capacity, hierarchically managed storage

Historical Perspective

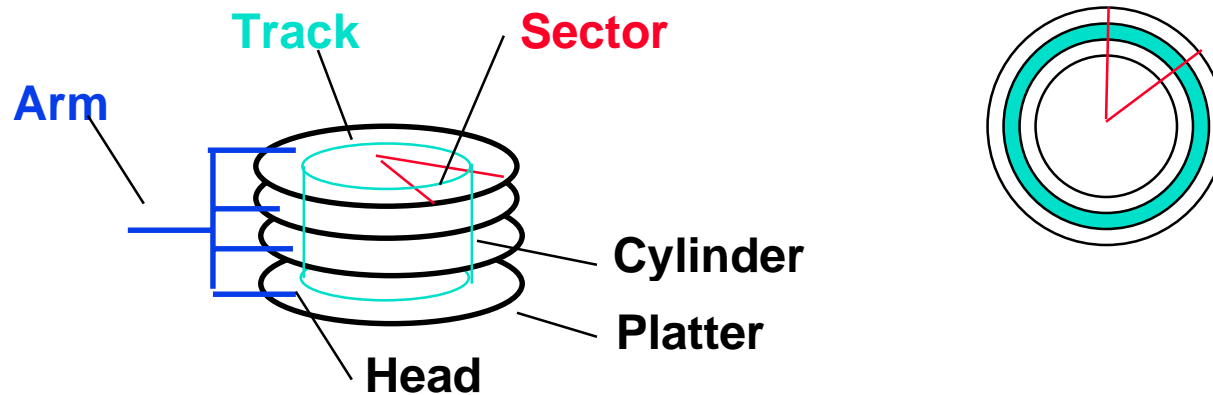


Types of Storage Devices

- **Magnetic Disks**
- **Magnetic Tapes**
- **CD ROM**
- **Juke Box (automated tape library, robots)**

Magnetic Disks

- Long term nonvolatile storage
- Another slower, less expensive level of memory hierarchy



Disk Access

- **Access time =**

queue + seek + rotational + transfer + overhead

- **Seek time**

- * move arm over track
- * average is confusing (startup, slowdown, locality of accesses)

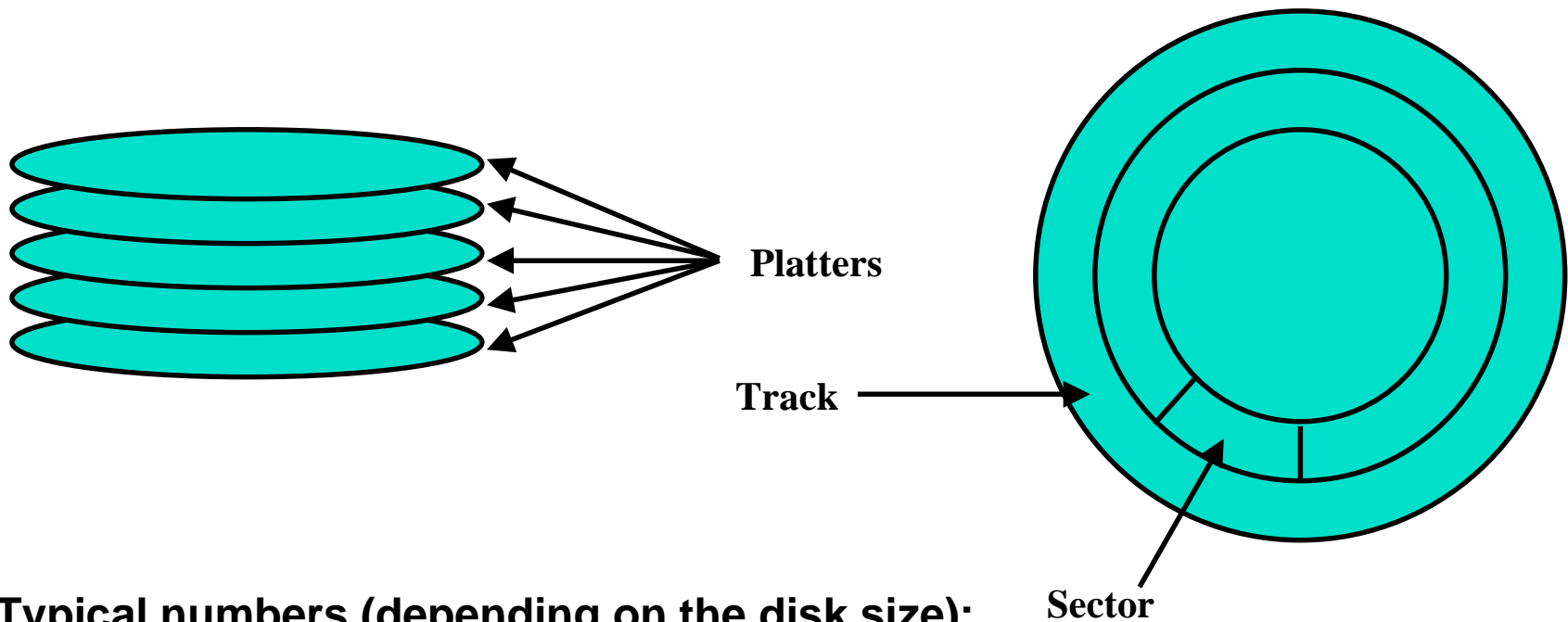
- **Rotational latency**

- * wait for sector to rotate under head
- * average = $0.5 / (3600 \text{ RPM}) = 8.3 \text{ms}$

- **Transfer Time**

- * $f(\text{size, BW bytes/sec})$

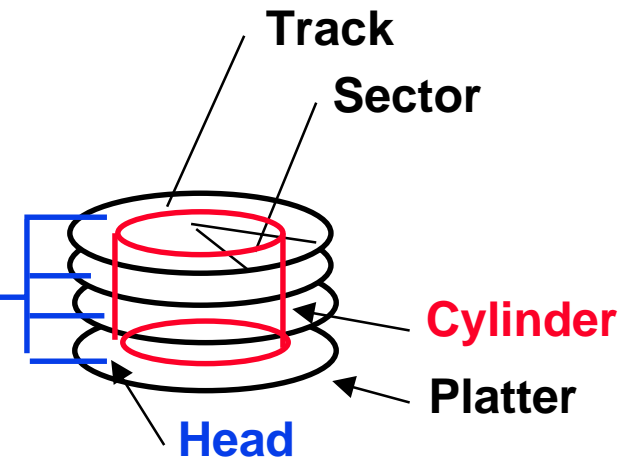
Organization of a Hard Magnetic Disk



- **Typical numbers (depending on the disk size):**
 - * 500 to 2,000 tracks per surface
 - * 32 to 128 sectors per track
 - A sector is the smallest unit that can be read or written
- **Traditionally all tracks have the same number of sectors:**
 - * **Constant bit density:** record more sectors on the outer tracks
 - * **Recently relaxed:** constant bit size, speed varies with track location

Magnetic Disk Characteristic

- **Cylinder:** all tracks under the heads at a given arm position (all tracks of same radius)

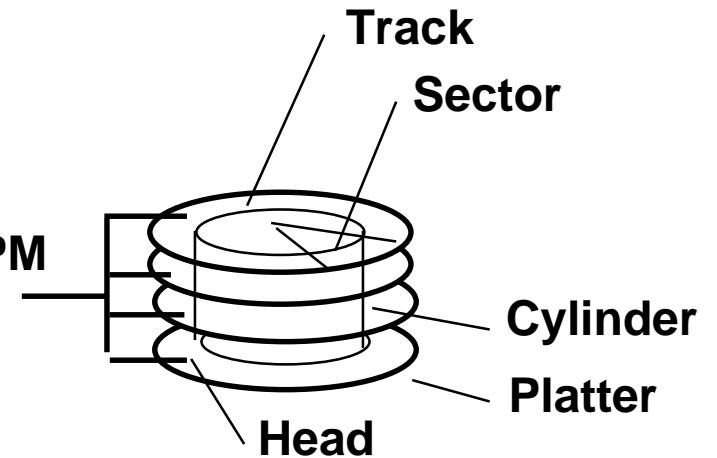


- **Read/write data is a three-stage process:**
 - * **Seek time:** position the arm over the proper track
 - * **Rotational latency:** wait for the desired sector to rotate under the read/write head
 - * **Transfer time:** transfer a block of bits (sector) under the read-write head
- **Average seek time as reported by the industry:**
 - * Typically in the range of 8 ms to 12 ms
 - * $(\text{Sum of the time for all possible seek}) / (\text{total \# of possible seeks})$
- **Due to locality of disk reference, actual average seek time may:**
 - * Only be 25% to 33% of the advertised number

Typical Numbers of a Magnetic Disk

- **Rotational Latency:**

- * **Most disks rotate at 3,600 to 10,000 RPM**
- * **Approximately 16 ms to 3.5 ms per revolution, respectively**
- * **An average latency to the desired information is halfway around the disk: 8 ms at 3600 RPM, 4 ms at 7200 RPM**



- **Transfer Time is a function of :**

- * **Transfer size (usually a sector): 1 KB / sector**
- * **Rotation speed: 3600 RPM to 7200 RPM**
- * **Recording density: bits per inch on a track**
- * **Diameter typical diameter ranges from 2.5 to 5.25 in**
- * **Typical values: 2 to 12 MB per second**

Tape vs. Disk

- Longitudinal tape uses same technology as hard disk; tracks its density improvements
- Inherent cost-performance based on geometries:
fixed rotating platters with gaps
(random access, limited area, fixed media)

vs.

removable long strips wound on spool
(sequential access, "unlimited" length)

- New technology trend:
Helical Scan (VCR, Camcoder, DAT)
Spins head at angle to tape to improve density

Storage costs

Media	Capacity	Cost	\$/byte
Disk	4-20GB	\$100-\$1000	2×10^{-8}
Tape	2-70GB	\$4-\$100	1×10^{-9}
CDR	0.75GB	\$1	1.4×10^{-9}
Flash EPROM	2MB	\$10	5×10^{-5}
Paper	10KB	\$0.01	1×10^{-6}

CDR vs. Tape

	CDR	Helical Scan Tape
Type	5.25"	8mm
Capacity	0.75 GB	5-12 GB
Media Cost	\$2	\$8
Drive Cost	\$400	\$800
Access	Write Once	Read/Write
Robot Time	10 - 20 s	10 - 20 s

Media cost ratio CDR vs. helical tape
= 4 : 1

Current Drawbacks to Tape

- **Tape wear out:**
 - Helical 100s of passes to 1000s for longitudinal
- **Head wear out:**
 - 2000 hours for helical
- **Both must be accounted for in economic / reliability model**
- **Long rewind, eject, load, spin-up times;**