# Computer Systems and Networks

ECPE 170 – Jeff Shafer – University of the Pacific

## Input / Output

## Schedule

- **Quiz 6** – Tuesday, Nov 22nd
  - Input / Output
  - Operating Systems
  - Compilers & Assemblers

## Input / Output

## I/O and Performance

- **Starting Chapter 7**
- Data storage and retrieval is one of the primary functions of computer systems
- Sluggish I/O throughput can have a ripple effect, dragging down overall system performance
  - This is especially true when virtual memory is involved
- The fastest processor in the world is of little use if it spends most of its time waiting for data
- If we really understand what's happening in a computer system we can make the best possible use of its resources

## Amdahl's Law

- The overall performance of a system is a result of the interaction of all of its components
- System performance is most effectively improved when the performance of the most heavily used components is improved
- This idea is quantified by Amdahl's Law:

$$S = \frac{1}{(1-f) + \frac{f}{k}}$$

where *S* is the overall speedup;
*f* is the fraction of work performed by a faster component; and
*k* is the speedup of the faster component

## Amdahl’s Law

- Amdahl’s Law can estimate the performance improvement of upgrading a system component
- On a large system, suppose we can upgrade a CPU to make it 50% faster for $10,000 or upgrade its disk drives for $7,000 to make them 150% faster
- Processes spend 70% of their time running in the CPU and 30% of their time waiting for disk service
- **An upgrade of which component would offer the greater benefit for the lesser cost?**

## Amdahl's Law

- The processor option offers a 30% speedup:

$$f = 0.70, \quad k = 1.5 \qquad S = \frac{1}{(1 - 0.7) + 0.7/1.5}$$

- And the disk drive option gives a 22% speedup:

$$f = 0.30, \quad k = 2.5 \qquad S = \frac{1}{(1 - 0.3) + 0.3/2.5}$$

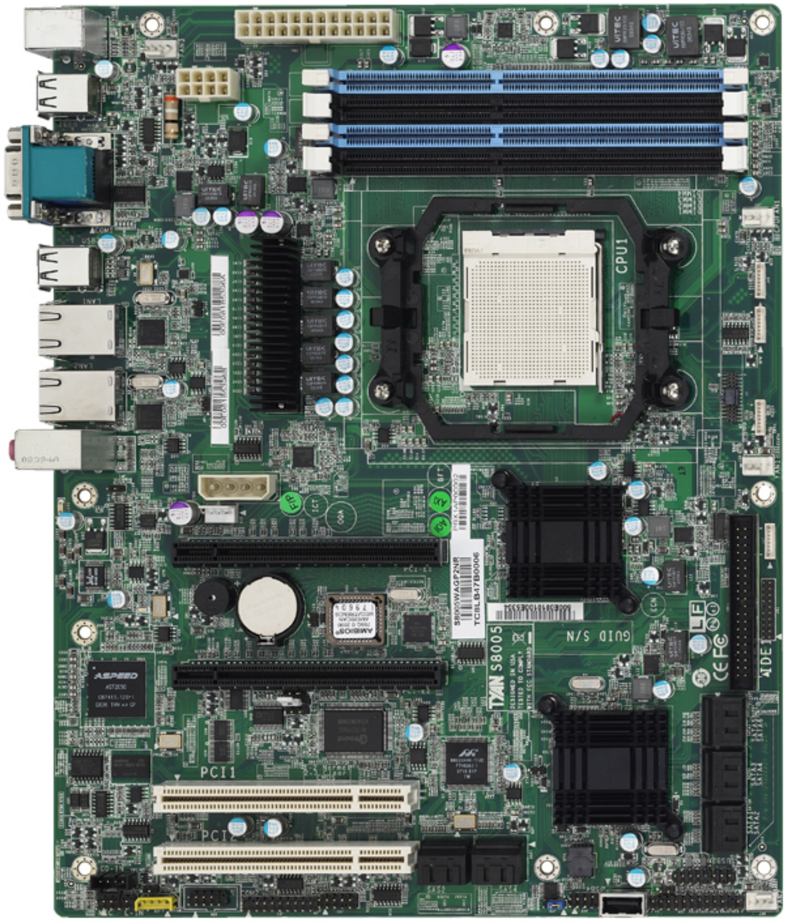
- Each 1% of improvement for the processor costs \$333, and for the disk a 1% improvement costs \$318

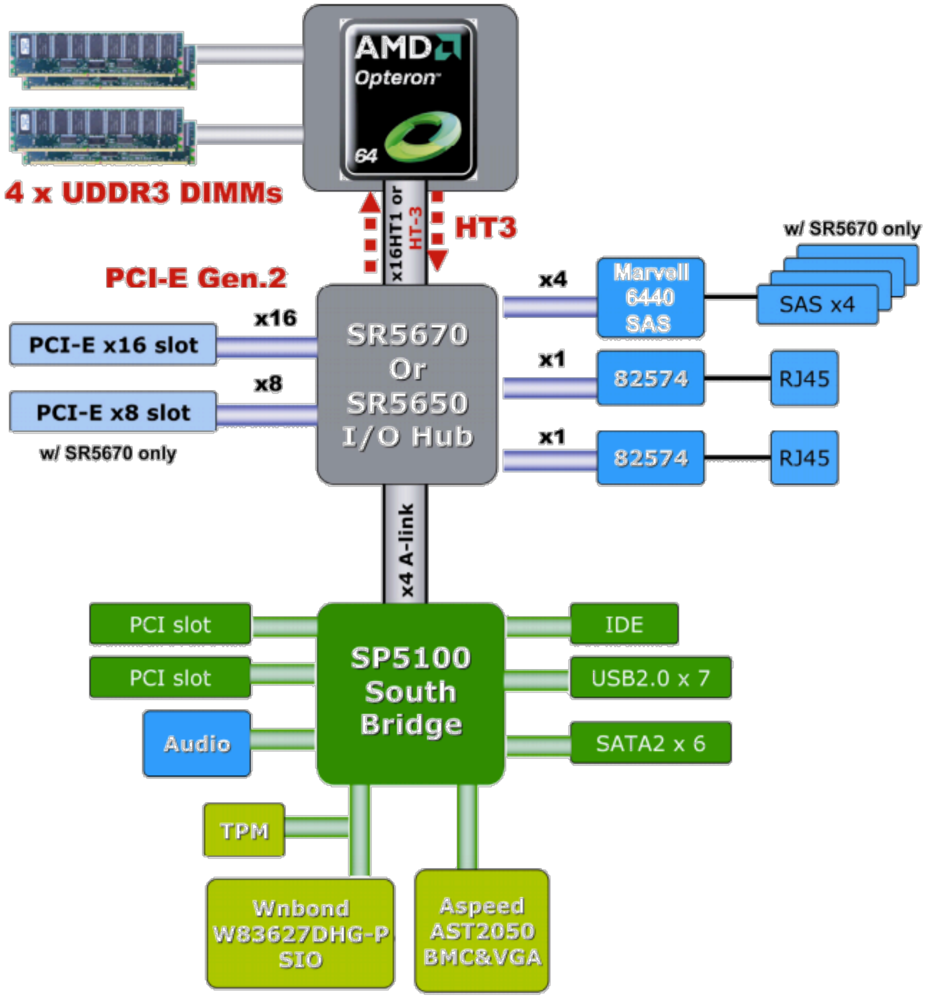
## I/O Architecture

## I/O Architectures

- Definition of I/O subsystem: components that move data between external devices and a host system
- I/O subsystems include:
  - **Blocks of main memory** that are devoted to I/O functions
  - **Buses** that move data into and out of the system.
  - **Control modules** in the host and in peripheral devices
  - **Interfaces to external component**s such as keyboards and disks
  - **Cabling or communications links** between the host system and its peripherals

# Modern AMD Opteron System





## I/O Architectures

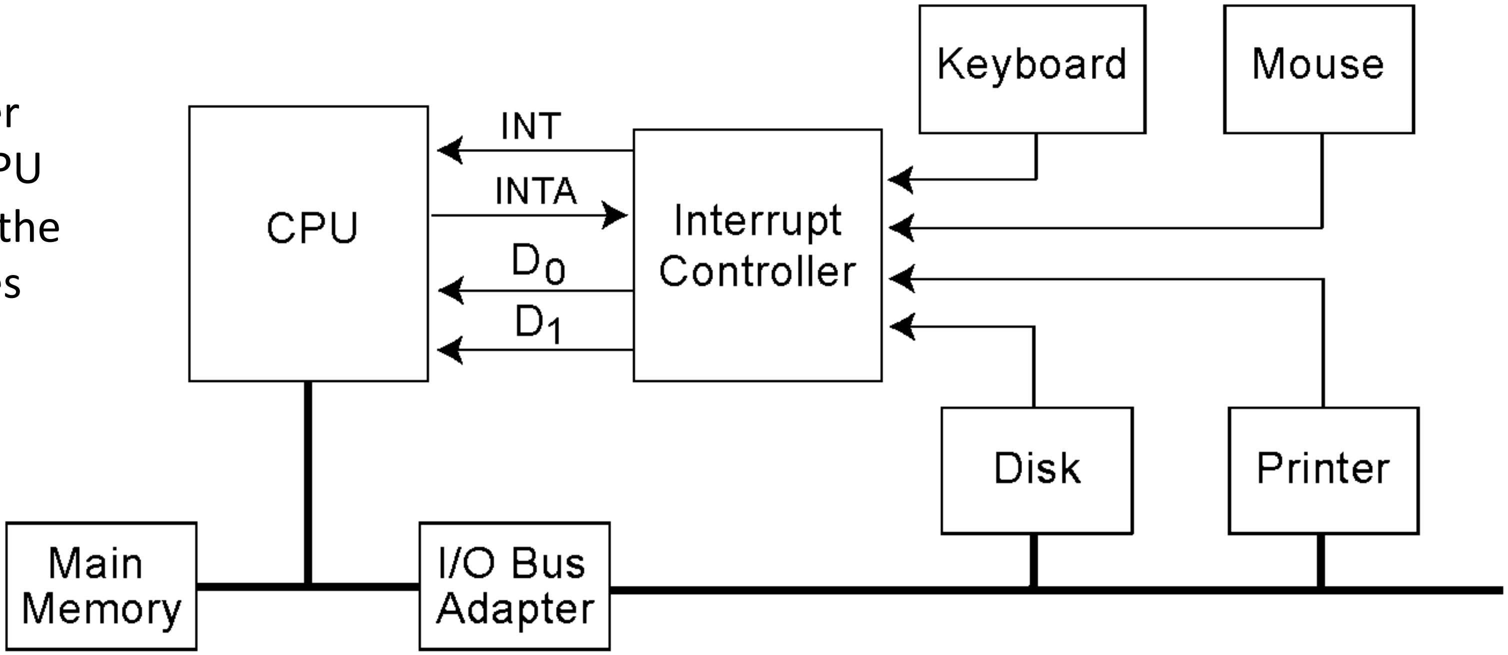
- **Programmed I/O**
  - Reserves a register for each I/O device
  - Each register is continually polled in software to detect data arrival
- **Interrupt-Driven I/O**
  - Allows the CPU to do other things until I/O is requested
- **Memory-Mapped I/O**
  - Shares memory address space between I/O devices and program memory
- **Direct Memory Access (DMA)**
  - Offloads I/O processing to a special-purpose chip that takes care of the details.

***These are not mutually exclusive categories!***

## I/O Architectures – Interrupts

- Each device connects its interrupt line to the interrupt controller.

The controller signals the CPU when any of the interrupt lines are asserted.



## I/O Architectures – Interrupts

- Interrupt signal is checked at the top of the fetch-decode-execute cycle
  - Interrupt? Save system state, go run **interrupt service routine,** and restore system state afterwards
  - No interrupt – continue
- **Interrupt service routine** - The specific subroutine that is executed whenever a specific interrupt occurs
  - Subroutine chosen by set of addresses (called **interrupt vectors**)

## I/O Architectures – Memory-Mapped I/O

- In memory-mapped I/O, devices and main memory share the same address space
  - Each I/O device has its own reserved block of memory
  - Memory-mapped I/O is just CPU memory accesses
    - The same instructions move data to and from both I/O and memory – simple system design!
- In *memory-mapped I/O*, the **CPU** is initiating the memory transfers
- In *direct-memory access I/O*, a **DMA controller** (separate hardware element) is initiating the memory transfers

## Character versus Block I/O

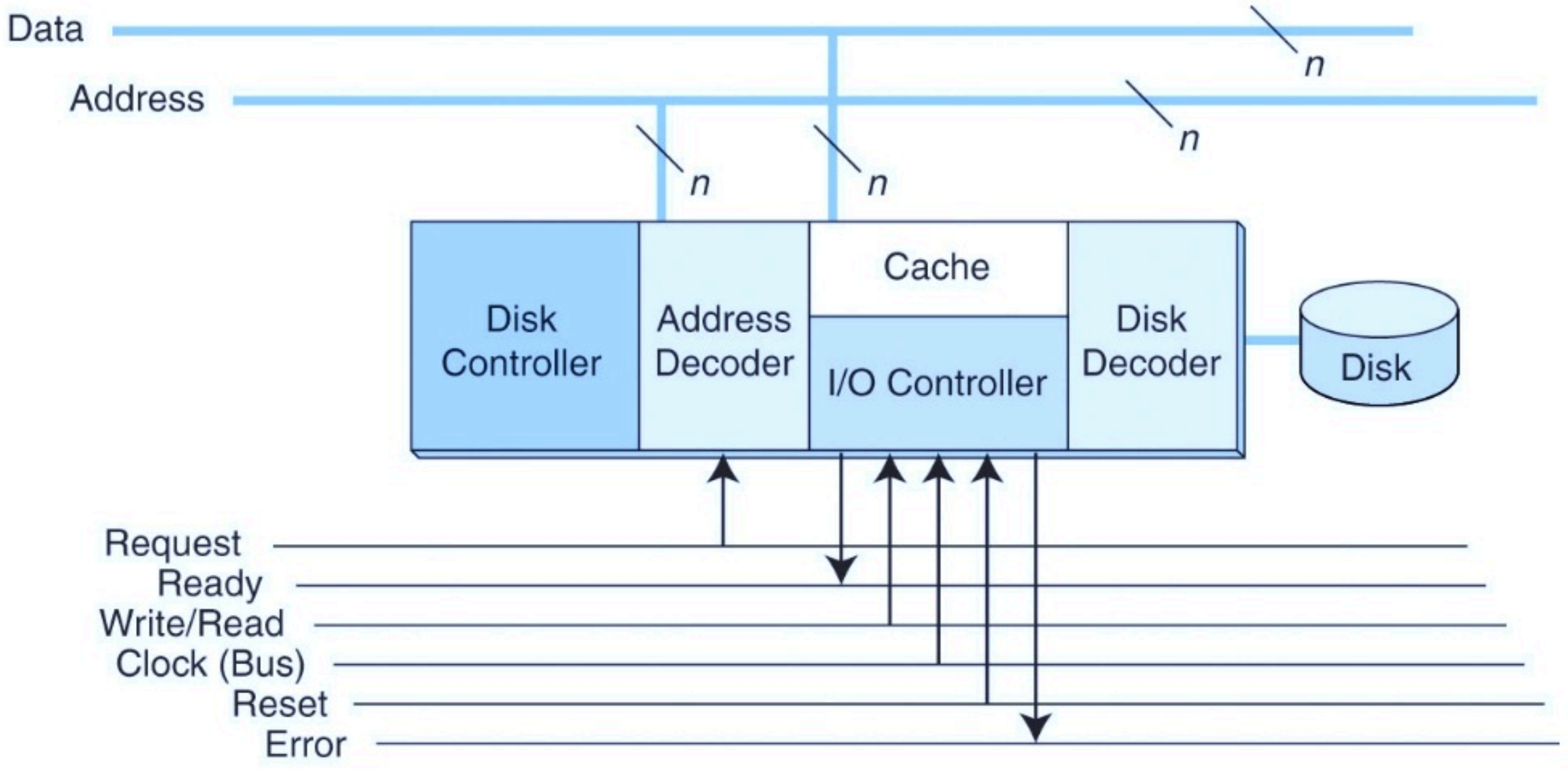
- Character I/O devices process one byte (or character) at a time
  - Legacy devices! PS/2 keyboards, mice, modems, etc…
  - Usually connected through an interrupt-driven I/O system
- Block I/O devices handle bytes in larger groups
  - Disks, network cards, video cards, etc…
  - Most efficiently connected through DMA

## I/O Bus

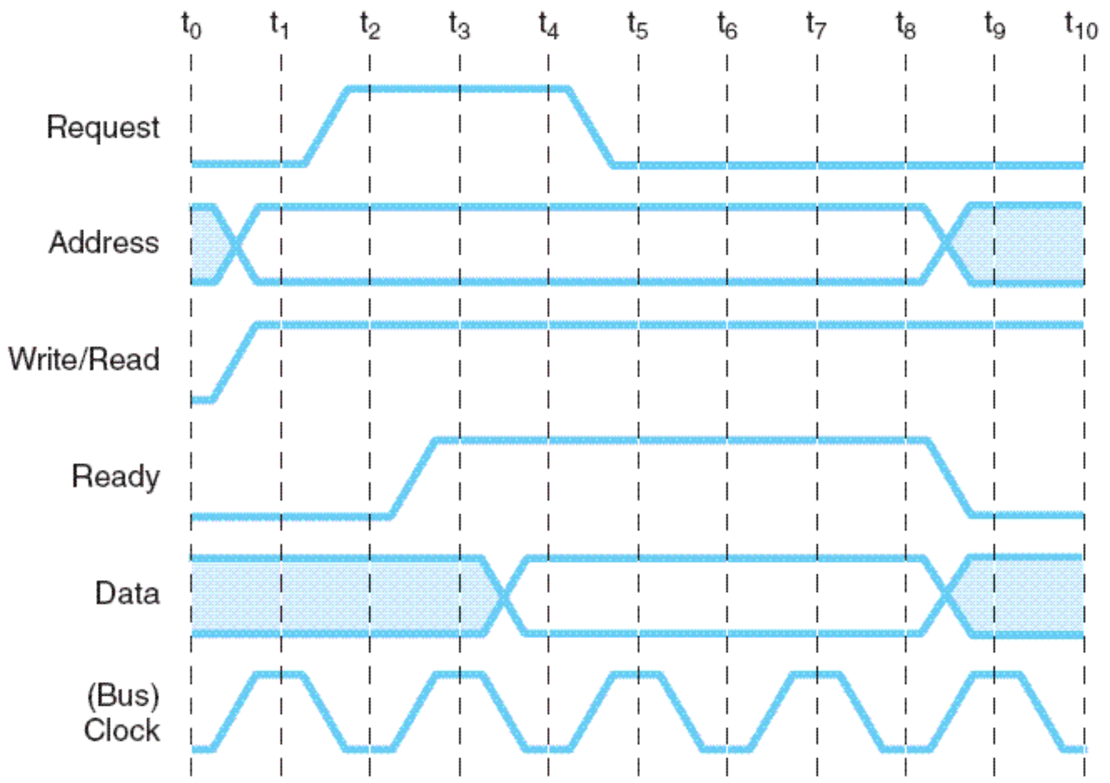
- Similar to memory bus – group of wires in parallel
  - The number of data lines is the width of the bus
- Key difference – Devices on I/O bus operate asynchronously!
  - Requests for bus access must be **arbitrated** among the devices involved
- **Bus control lines** activate the devices when they are needed, raise signals when errors have occurred, and reset devices when necessary
- **Bus clock** coordinates activities and provides bit cell boundaries

## I/O Architectures

- Example: Bus connected to a disk drive



- Timing diagrams define bus operation in detail.



| Time | Salient Bus Signal | Meaning |
|---|---|---|
| $t_0$ | Assert Write | Bus is needed for writing (not reading) |
| $t_1$ | Assert Address | Indicates where bytes will be written |
| $t_2$ | Assert Request | Request write to address on address lines |
| $t_3$ | Assert Ready | Acknowledges write request, bytes placed on data lines |
| $t_4$–$t_7$ | Data Lines | Write data (requires several cycles) |
| $t_8$ | Lower Ready | Release bus |

## Data Transmission Modes

- **Parallel**
  - Interface requires one conductor for each bit
    - i.e. with 8 wires in parallel, we can move an entire byte at once
- **Serial**
  - Multiple bits are multiplexed onto a single conductor (and demultiplexed at other side)
  - Increasingly popular
    - Less problems with *clock skew* between parallel wires
    - Less susceptible to attenuation / interference
    - Fewer wires (and pins) simplify circuit board and chip designs

## Legacy Technologies

## Legacy Technologies

- "Legacy" doesn't mean that no one uses them anymore, or that that aren't competitive in some industries
- Optical Disks? (Section 7.7)
- Magnetic Tape? (Section 7.8)
  - Up to 1.45TB per tape with Generation 5 "Linear Tape Open" standard, released in 2010

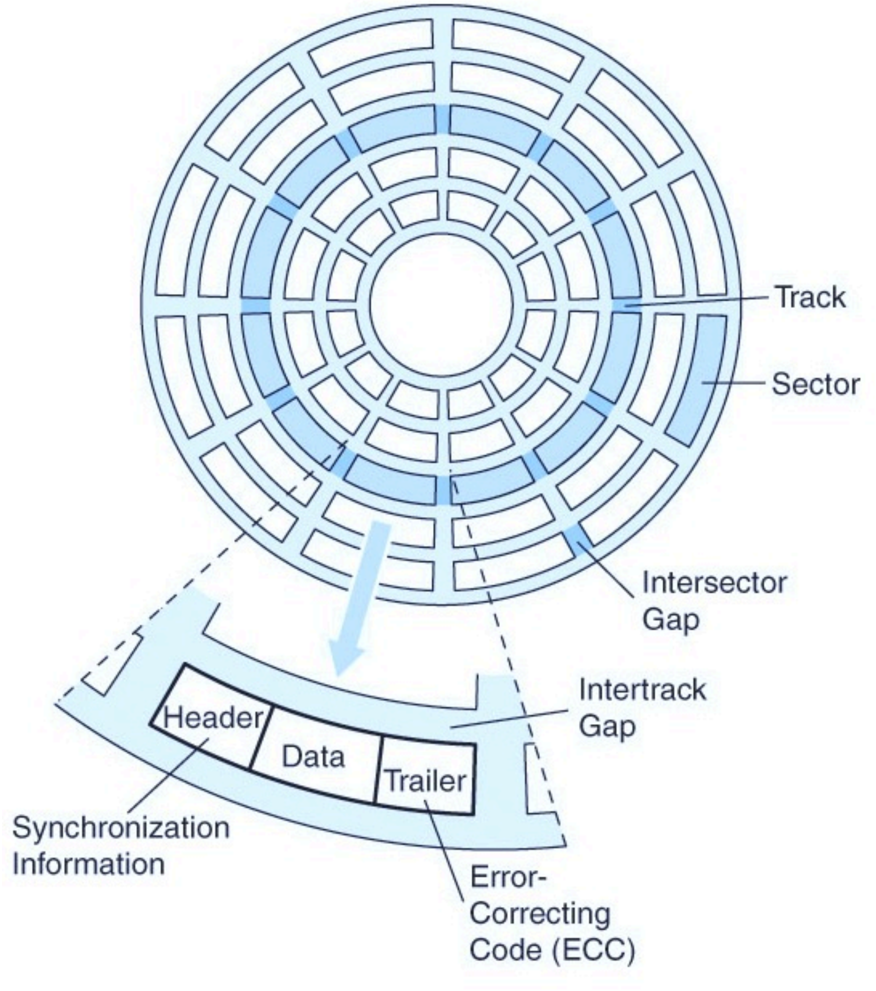
## Magnetic Disks

## Magnetic Disk Technology

- Magnetic disks offer large amounts of durable storage that can be accessed quickly
- Disk drives are called random (or direct) access storage devices, because blocks of data can be accessed according to their location on the disk
  - This term was coined when all other durable storage (e.g., tape) was sequential

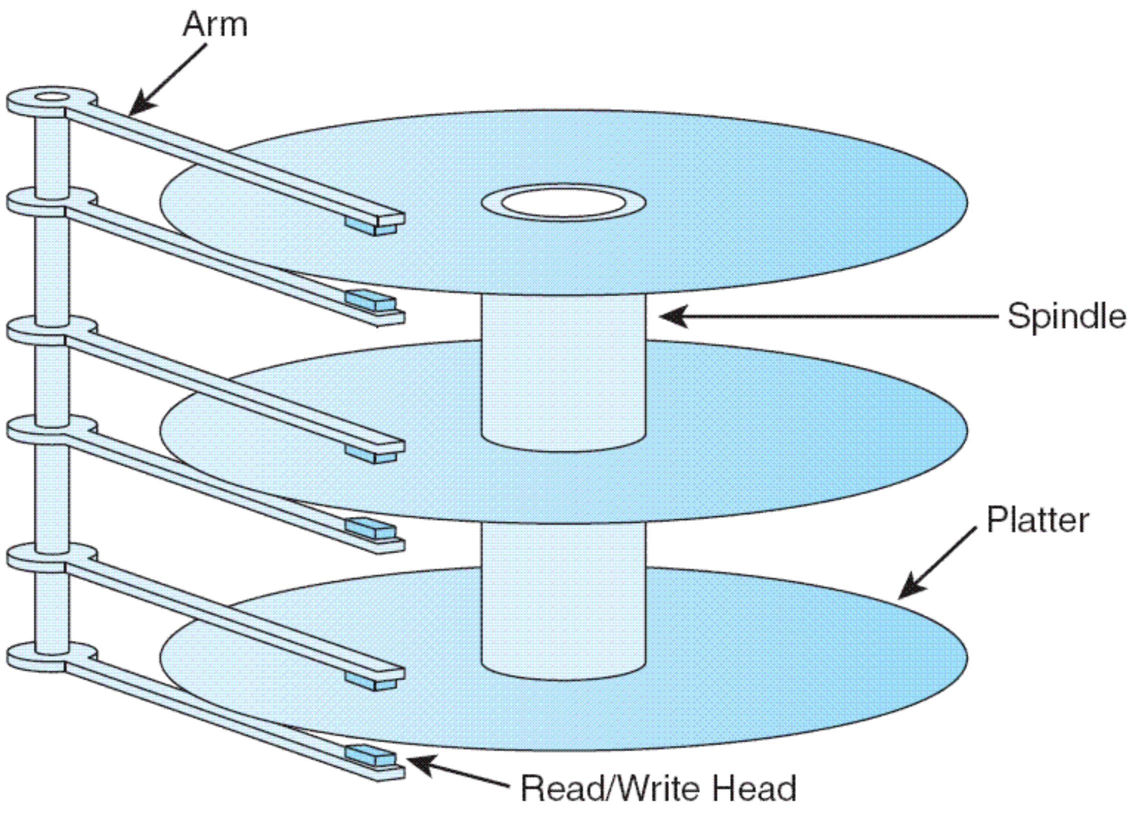
## Magnetic Disk Technology

- Disk **tracks** are numbered from the outside edge, starting with zero
  - The track is the entire ring
- A **sector** is one portion of a track!



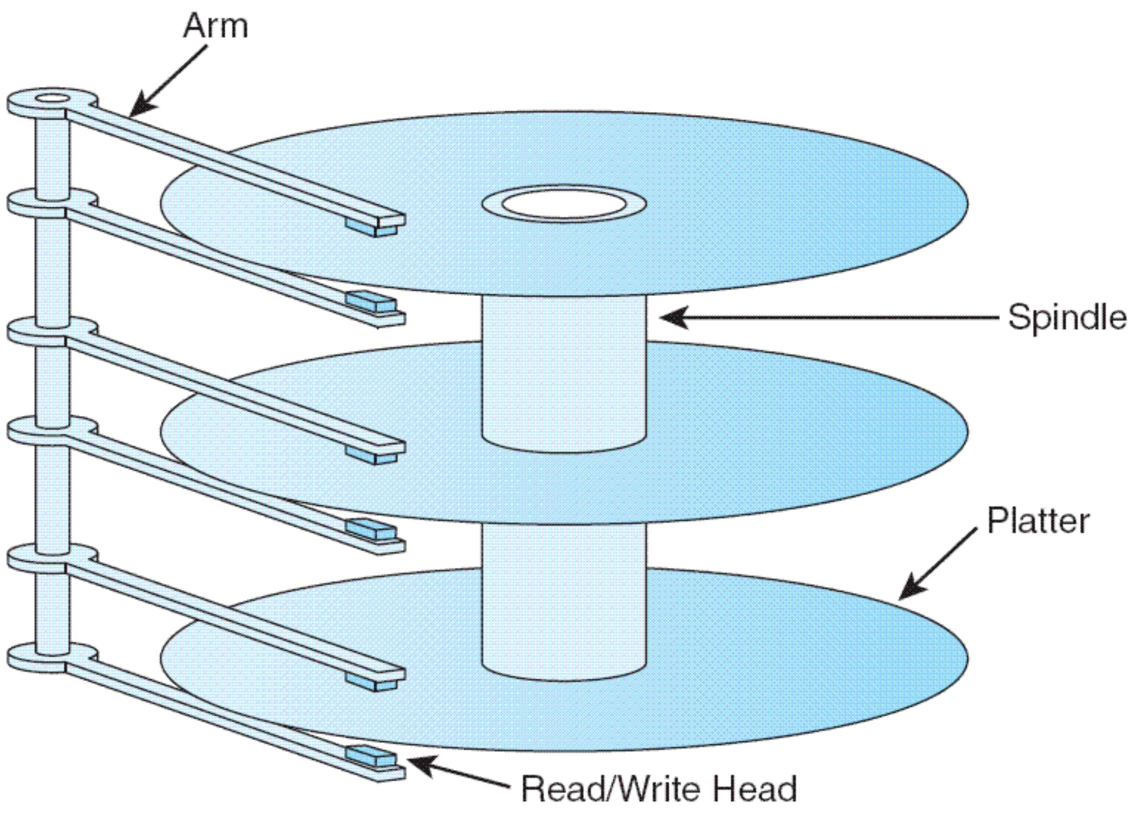
## Magnetic Disk Technology

- Hard disk platters are mounted on spindles
- Read/write heads are mounted on a comb that swings radially to read the disk



## Magnetic Disk Technology

- The rotating disk forms a logical **cylinder** beneath the read/write heads
- Data blocks are addressed by their **cylinder**, **surface**, and **sector**



## Magnetic Disk Technology

- There are a number of *electromechanical* properties of hard disk drives that determine how fast its data can be accessed
- **Seek time** – time that it takes for a disk arm to move into position over the desired cylinder
- **Rotational delay** – time that it takes for the desired sector to move into position beneath the read/ write head
- Seek time + rotational delay = **access time**

# Magnetic Disk Technology

- **Transfer rate** – rate at which data can be read from the disk.
- **Average latency** - function of the rotational speed:

$$\frac{\dfrac{60\ \text{seconds}}{\text{disk rotation speed}} \times \dfrac{1000\ \text{ms}}{\text{second}}}{2}$$

- **Mean Time To Failure (MTTF)** – calculated via statistics from much shorter experiments
  - Limited guarantee – your own disks could vary significantly!

## Magnetic Disk Technology

- Exercise – Suppose a disk drive has these characteristics:
  - 4 surfaces
  - 600 tracks/surface
  - 2000 sectors/track
  - 1024 bytes/sector
- **What is the capacity of the drive?**
  - 4 surfaces * 600 tracks/surface * 2000 sectors/track * 1024 bytes/sector = 4,915,200,000 bytes (=4.9 GB)
  - *Powers of 10 for hard drives, not 2!*

## Magnetic Disk Technology

- Exercise – Suppose a disk drive has these characteristics:
  - 8ms track-to-track seek time (average)
  - 7200 RPM rotational speed
- **What is its access time?**
  - Access time = Seek latency + rotational latency
  - 7200 rev/min → 120 rev/sec = 0.12 rev/ms
  - 0.12 rev/ms → 8.333 ms/rev
  - 8 ms (seek) + ½ * 8.333 ms (rev) = 12.167 ms

## How Big Will Hard Drives Get?

- Advances in technology have defied all efforts to define the ultimate upper limit for magnetic disk storage
  - In the 1970s, the upper limit was thought to be around 2Mb/in$^2$
- As data densities increase, bit cells consist of proportionately fewer magnetic grains
  - There is a point at which there are too few grains to hold a value, and a 1 might spontaneously change to a 0, or vice versa
  - This point is called the **superparamagnetic limit**

## How Big Will Hard Drives Get?

- **When will the limit be reached?**
- In 2006, the limit was thought to lie between 150Gb/in$^2$ and 200Gb/in$^2$ *(with longitudinal recording technology)*
- In 2010, commercial drives have densities up to 667Gb/in$^2$
- In 2010, predicted drives have densities up to 1 Tbit/in$^2$ (1024 Gbit/in$^2$) *(with perpendicular recording)*

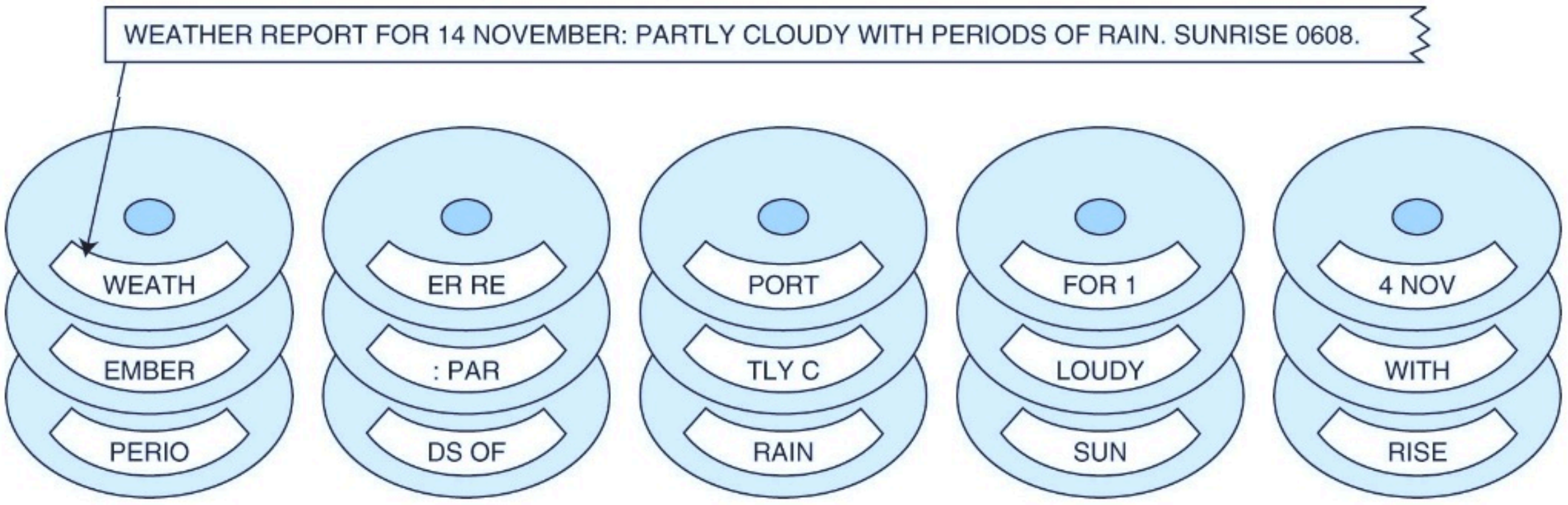
## RAID

## RAID

- **RAID - Redundant Array of Independent Disks**
  - Goals: Improved reliability, cost, and performance
- Data is stored across many disks (an array of disks)
  - Disks added to the array to provide error correction (redundancy)

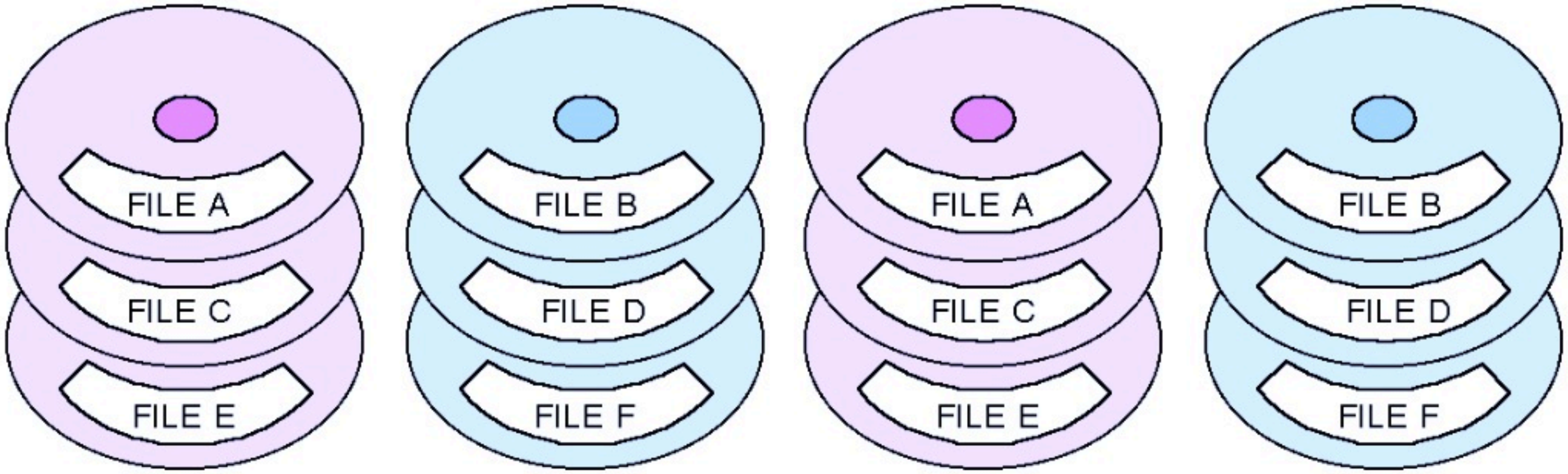
## RAID Level 0 - Spanning

- **RAID 0:** Improved performance, but no redundancy
  - Data is written in blocks across the entire array
  - Reliability is worse – **Why?**


WEATHER REPORT FOR 14 NOVEMBER: PARTLY CLOUDY WITH PERIODS OF RAIN. SUNRISE 0608.
WEATH
EMBER
PERIO
ER RE
: PAR
DS OF
PORT
TLY C
RAIN
FOR 1
LOUDY
SUN
4 NOV
WITH
RISE

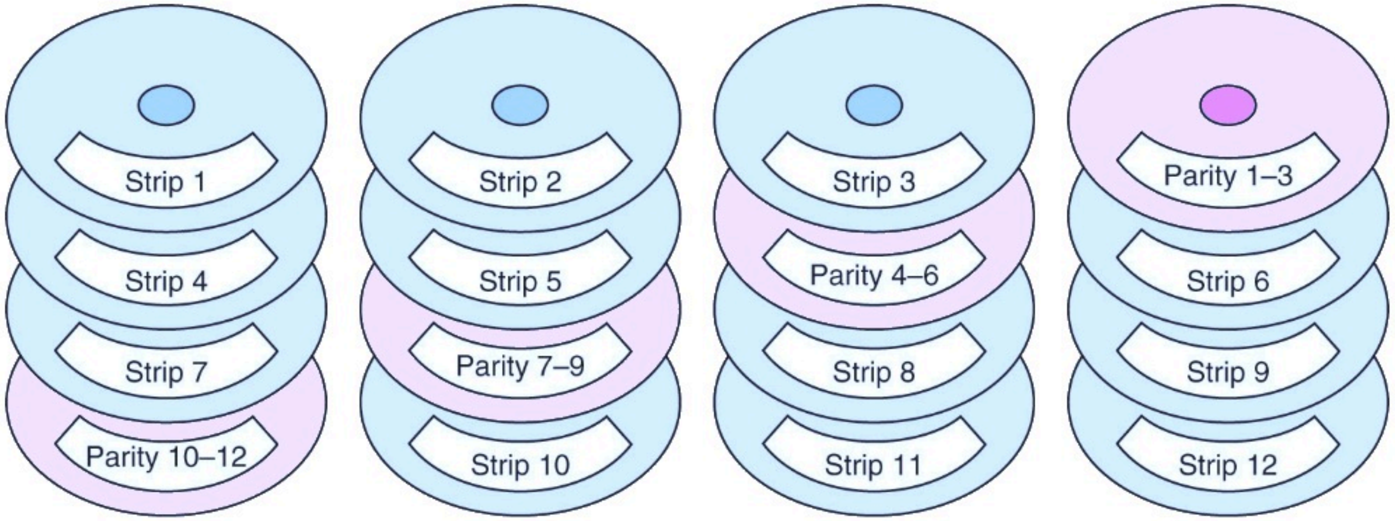
## RAID Level 1 - Mirroring

- **RAID 1:** 100% redundancy and good performance
  - Two matched sets of disks contain the same data
  - Disadvantage – Cost double!



## RAID Level 5 – Distributed Parity

- **RAID 5:** Distributed parity
  - Stripe blocks of data across disks
  - Parity (XOR) of data stripes is calculated and also distributed across all disks
    - Any 1 disk can fail, and no data is lost
  - Good mix of performance and reliability



PARITY 1 – 3 = (Strip 1) XOR (Strip 2) XOR (Strip 3)

## RAID Notes

- Many other RAID levels (variations on these themes)
- A higher RAID level does not necessarily mean a "better" RAID level
  - It all depends upon what the application / user needs
  - Some applications need *bandwidth* without high redundancy (i.e. "scratch" space for data processing, like file sorting)
  - Some applications need *high redundancy* over high bandwidth
  - Need both? Critical, high-throughput files can benefit from combining RAID 0 with RAID 1, called RAID 10

## RAID Example

- Example – If you have ten 500GB disk drives, how much storage space (in GB) do you really get using:
  - RAID 0
  - RAID 1
  - RAID 5

## RAID Example

- RAID 0: 5TB (all disks are used, no redundancy)
- RAID 1: 2.5TB (one-to-one redundancy)
- RAIDs 5
  - If one parity drive, 9*0.5TB = 4.5TB
  - If one parity per four data, 2*4*0.5TB = 4TB

## Solid State Disks (SSD)

## Emergence of SSDs

- **Hard drive advantages?**
  - Low cost per bits
- **Hard drive disadvantages?**
  - Very slow compared to main memory
  - Fragile (ever dropped one?)
  - Moving parts wear out
- Reductions in flash memory cost is opening another possibility: **solid state drives** (SSDs)
  - SSDs appear like hard drives to the computer, but they store data in non-volatile **flash memory** circuits
  - Flash is **quirky!** Physical limitations pose engineering challenges…

## Flash Memory

- Typical flash circuits are built from dense arrays of NAND gates
- Different from hard drives – we can't read/write a single bit (or byte)
  - **Reading or writing?** Data must be read from an entire **flash page** (2kB-8kB)
    - Reading much faster than writing a page
    - It takes some time before the cell charge reaches a stable state
  - **Erasing?** An entire **erasure block** (32-128 pages) must be erased (set to all 1's) first before individual bits can be written (set to 0)
    - Erasing takes two orders of magnitude more time than reading

# Flash-based Solid State Drives (SSDs)

## Advantages

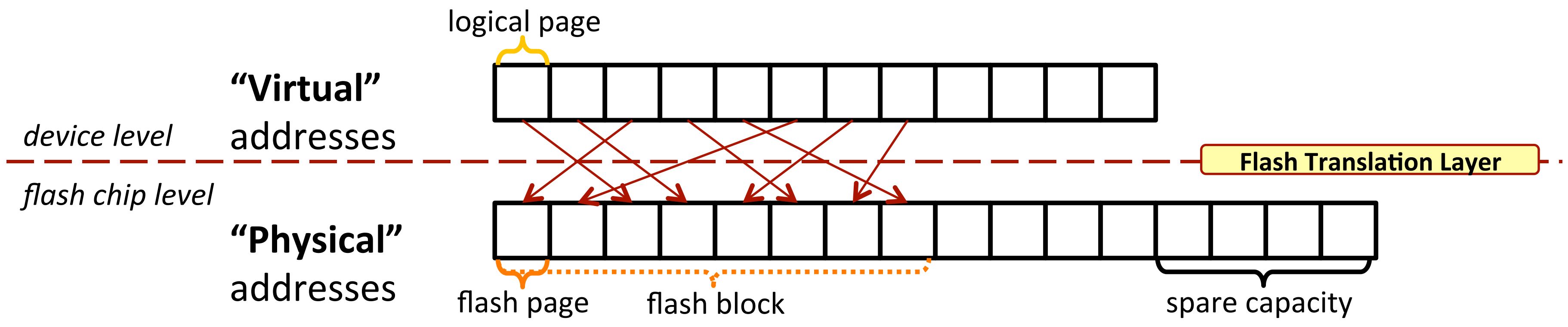
- Common I/O interface
  - Block-addressable interface
- No mechanical latency
  - Access latency is independent of the access pattern
    - Compare this to hard drives
- Energy efficient (no disk to spin)
- Resistant to extreme shock, vibration, temperature, altitude
- Near-instant start-up time

## Challenges

- Limited endurance and the need for **wear leveling**
- Very slow to erase blocks (needed before reprogramming)
  - Erase-before-write
- Read/write asymmetry
  - Reads are faster than writes

# Flash Translation Layer

- Solution to flash challenges?
- **Flash Translation Layer (FTL)**
  - **"Virtual" addresses** seen by the OS and computer
  - **"Physical" addresses** used by the flash memory
- Perform writes out-of-place
  - Amortize block erasures over many write operations
- Wear-leveling
  - Writing the same "virtual" address repeatedly won't write to the same physical flash location repeatedly!


logical page
"Virtual" addresses
device level
Flash Translation Layer
flash chip level
"Physical" addresses
flash page
flash block
spare capacity