Computer Systems and Networks

ECPE 170 – Jeff Shafer – University of the Pacific

Input / Output

Schedule

- **Today**
  - Chapter 7 – Input / Output Systems
- **Friday (March 30th)**
  - Input / Output Systems
  - **Quiz #5** – Chapter 6 (Cache & Virtual Memory)
- **Next Week**
  - Chapter 8 – Software
    - Operating systems, compilers, assemblers

Review

Cache Memory

- The hardware takes a (physical) memory address and divide it up into fields for use by the cache
- **What are the field names for a**
  - **Direct-map cache?**
  - **Set-associative cache?**
  - **Fully-associative cache?**

Cache Memory

- For a **direct-map cache**, I have the following fields
  - *Tag*
  - *Block*
  - *Offset*
- **How do I use these fields to search the cache for a given memory address?**
  - Go to *block* in cache. Does the *tag* match and valid bit set? If yes, retrieve data from *offset* byte in the cache block.

Cache Memory

- For a **fully-associative cache**, I have the following fields
  - *Tag*
  - *Offset*
- **How do I use these fields to search the cache for a given memory address?**
  - Search all blocks in cache. Does the *tag* match and valid bit set? If yes, retrieve data from *offset* byte in the cache block.

Cache Memory

- For a **set-associative cache**, I have the following fields
  - *Tag*
  - *Set*
  - *Offset*
- **How do I use these fields to search the cache for a given memory address?**
  - Search all blocks within specified *set in* cache (e.g. a 2-way set associative cache has 2 blocks in each set). Does the *tag* match and valid bit set? If yes, retrieve data from *offset* byte in the cache block.

HW #15 - 6.18

Page	Frame
0	**3**
4	1

TLB

	TAG	DATA	TAG	DATA
Set 0	00	C	01	I
Set 1	00	D	10	H

Cache

	Frame	Valid
0	3	1
1	0	1
2	-	0
3	2	1
4	1	1
5	-	0
6	-	0
7	-	0

Page Table

Frame		Block
0	C	0
	D	1
1	I	2
	J	3
2	G	4
	H	5
3	A	6
	B	7

Main Memory

Page		Block
0	A	0
	B	1
1	C	2
	D	3
2	E	4
	F	5
3	G	6
	H	7
4	I	8
	J	9
5	K	10
	L	11
6	M	12
	N	13
7	O	14
	P	15

Virtual Memory for Process P

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
  - 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

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 - **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 $1,000 or upgrade its disk drives for $700 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:

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

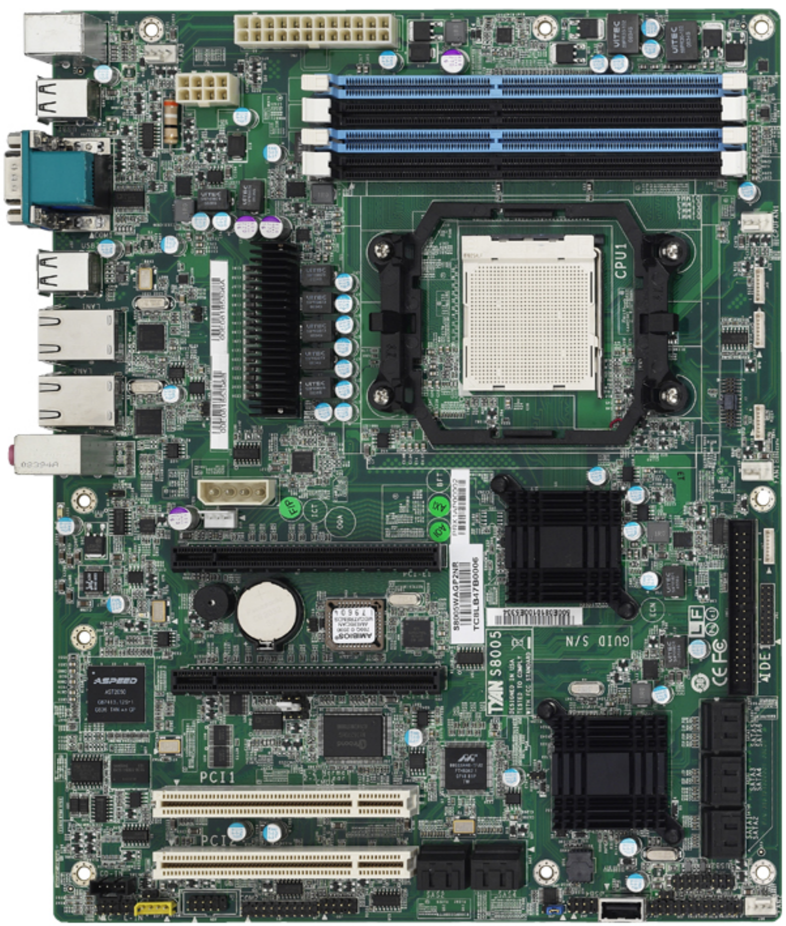
- Each 1% of improvement for the processor costs $33.33, and for the disk a 1% improvement costs $31.82

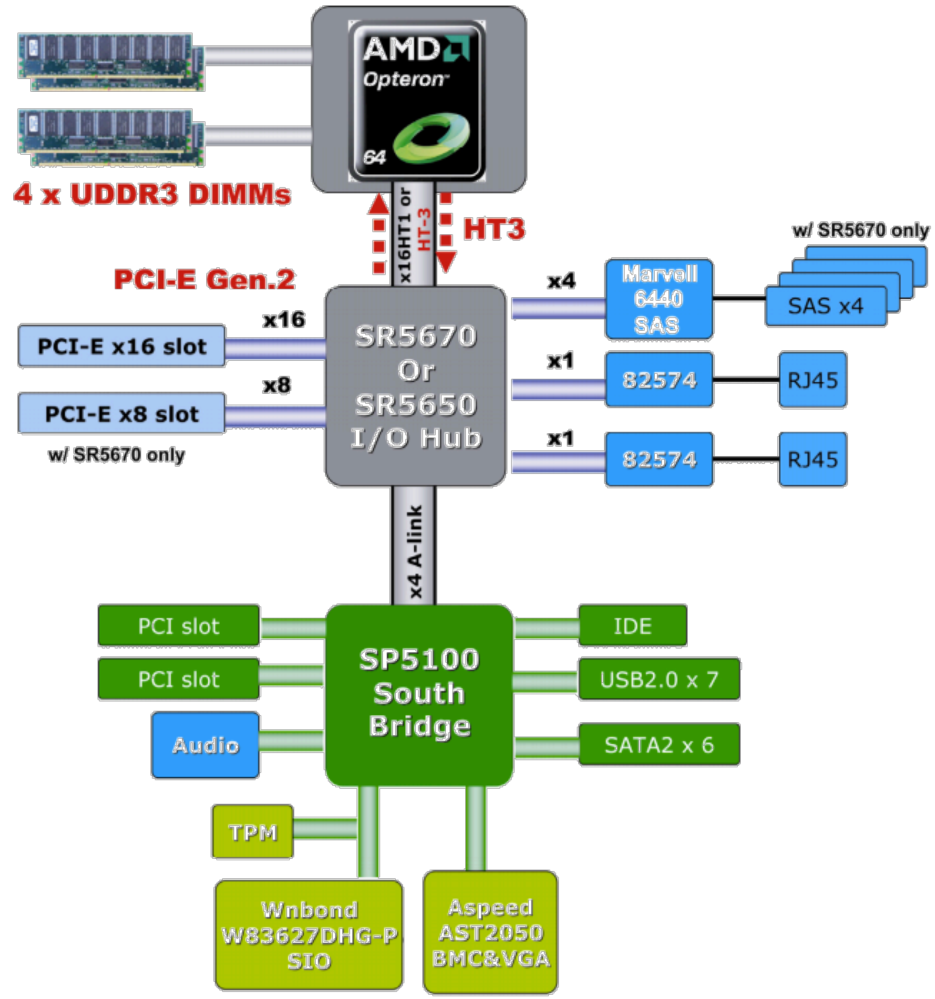
I/O Architecture

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

- 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

Data Transmission Modes

- **Parallel**
  - Interface requires one conductor for each bit
  - Example: 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 8TB per tape with Generation 6 "Linear Tape Open" standard, coming in 2012

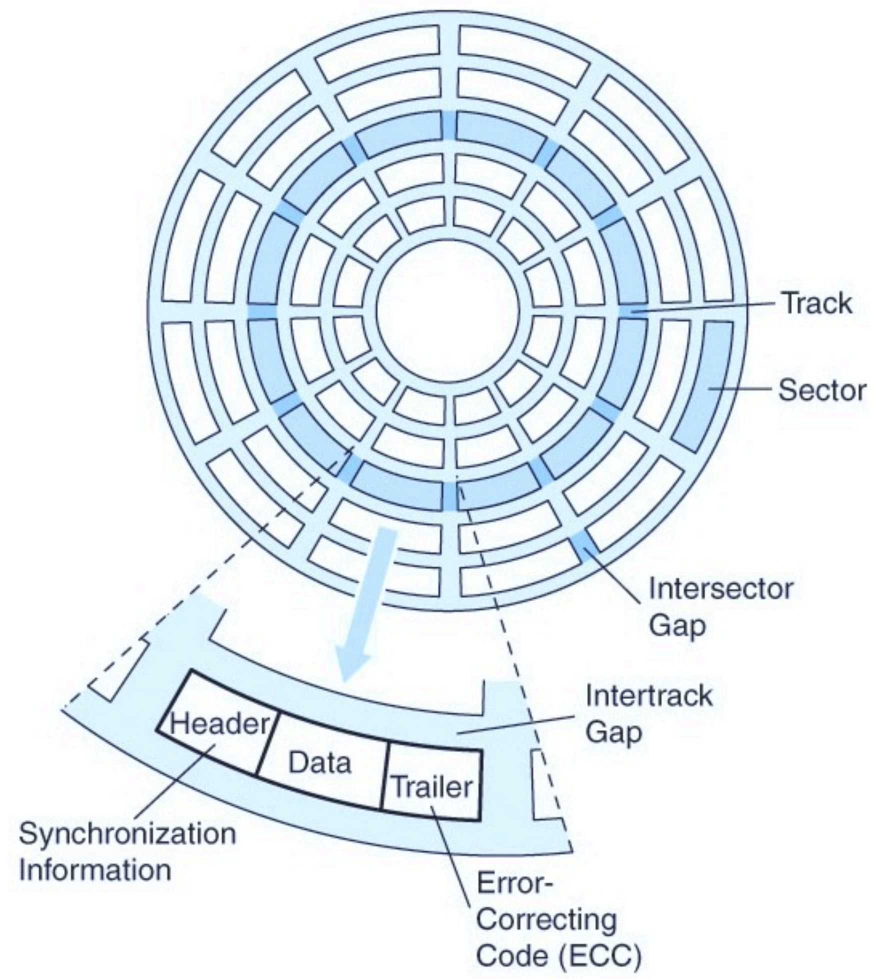
Magnetic Disks

Magnetic Disk Technology

- Magnetic disks offer large amounts of durable storage that can be accessed quickly
- Disk drives are called *random* 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

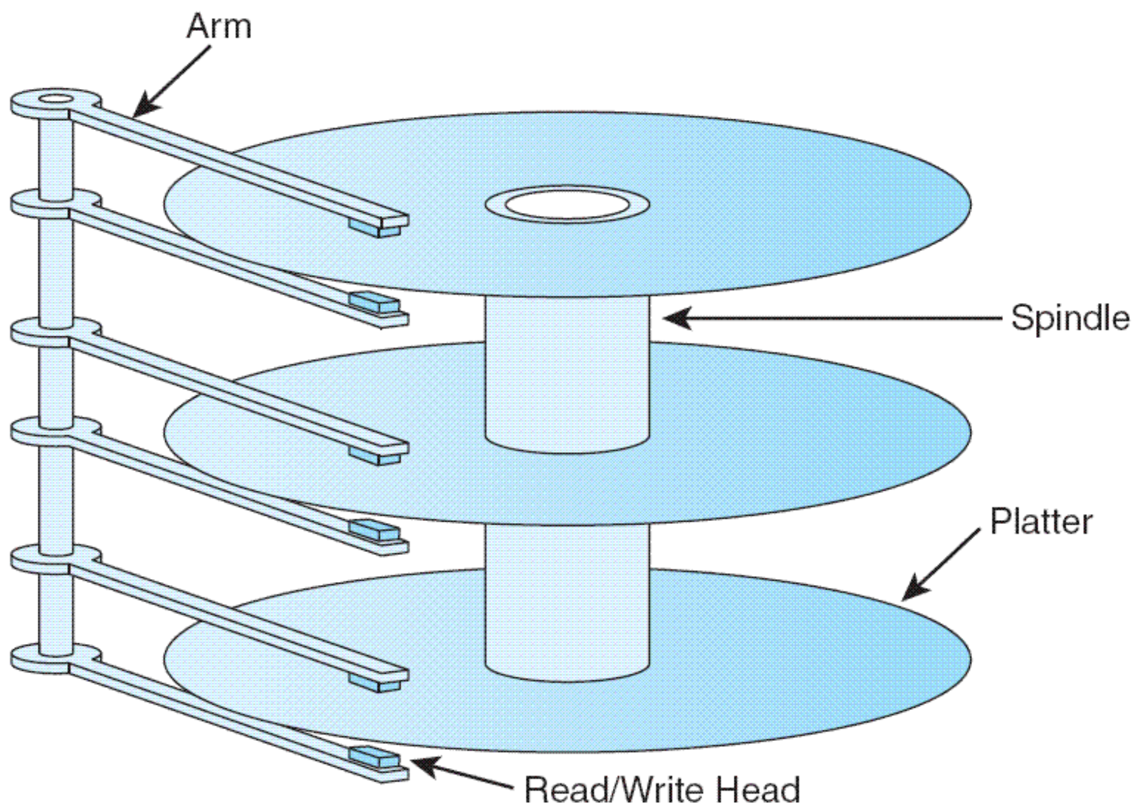
Magnetic Disk Technology

- 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!
- A **cylinder** is a single track, read across multiple disks



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
  - All heads move **together**!



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:
  - 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

↑ Average!

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)*
- 2010: Commercial drives have densities up to 667Gb/in^2
- 2012: Seagate demos drive with 1 Tbit/in^2 density
  - *With heat-assisted magnetic recording* – they use a laser to heat bits before writing
  - Each bit is ~12.7nm in length (a dozen atoms)

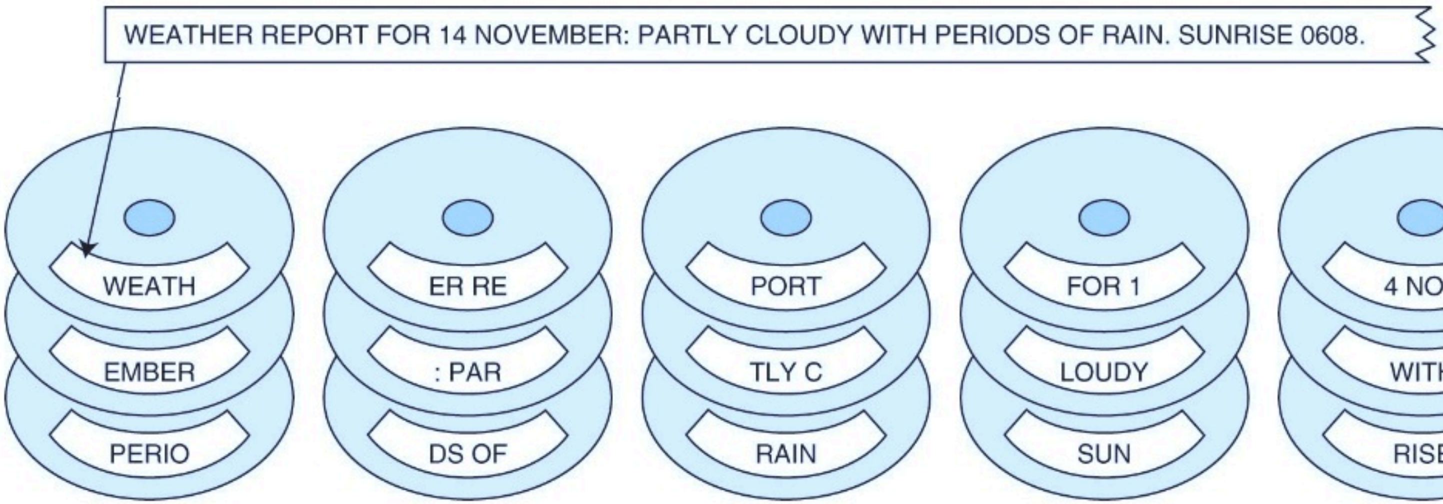
RAID

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)

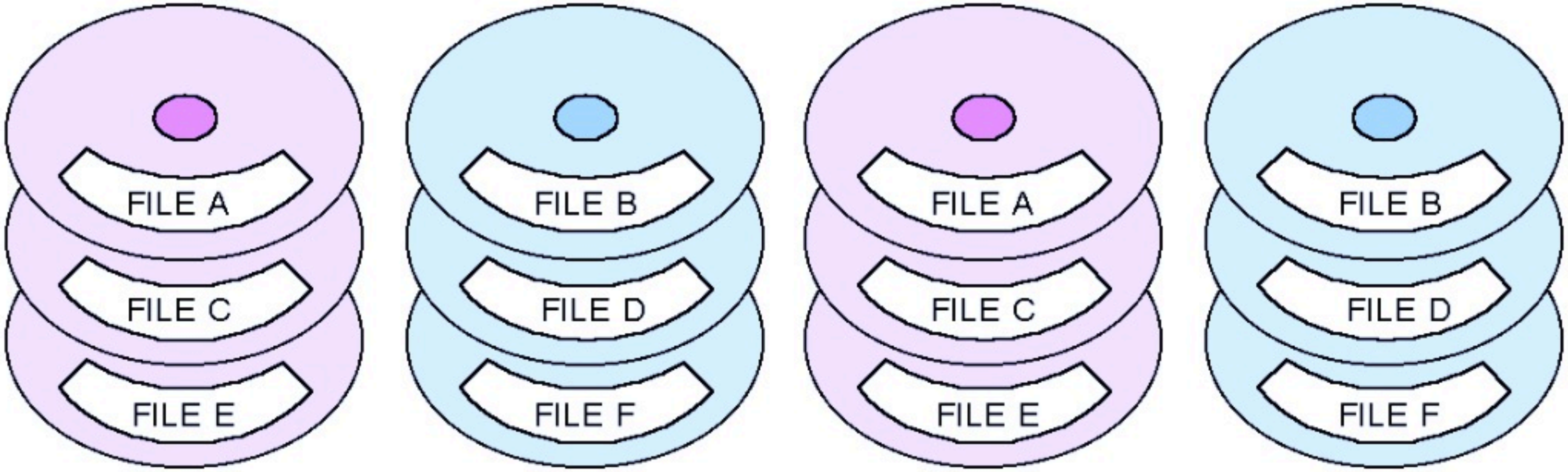
RAID Level 0 - Spanning

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



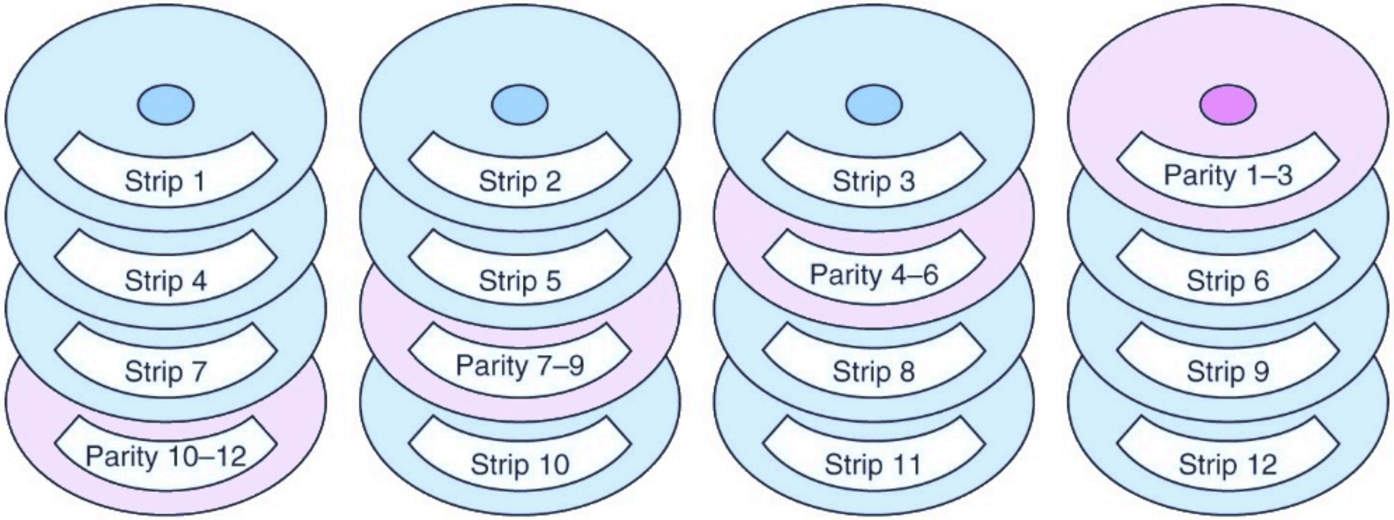
RAID Level 1 - Mirroring

- **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)
- RAID 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 chips 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

- Same block-addressable I/O interface as hard drives
- 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

- **Flash Translation Layer (FTL)**
  - Necessary for flash reliability and performance
  - **"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!