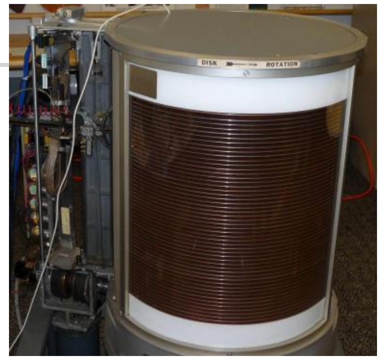
Chapter 3

COMPUTER MEMORY
Part 4
Storage devices

Data storage devices

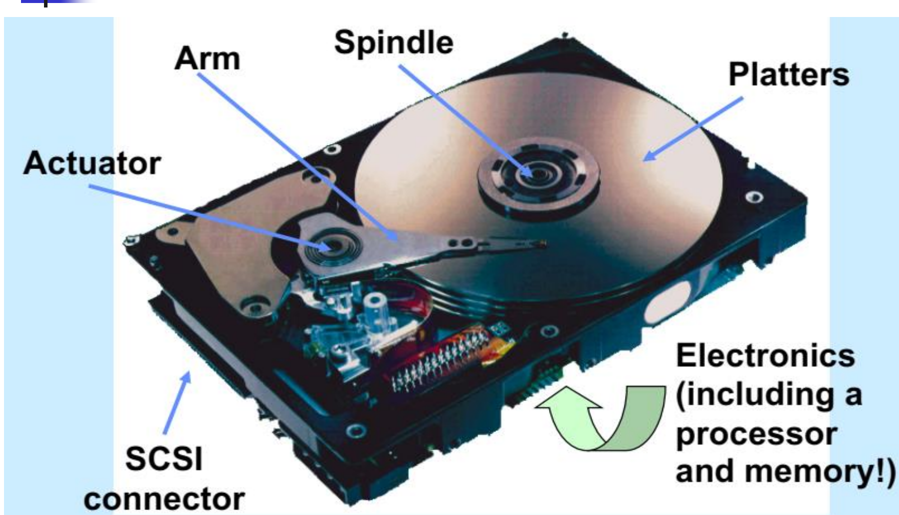
- All computers have data storage devices
- Their performance is important for the overall performance of the whole system
- They have a crucial role in virtual memory management
- We are going to cover:
  - HDD: Hard disk drives
  - SSD: Solid state drives
- There are others as well:
  - Optical drives: similar to HDDs at several aspects
  - Pendrives: are based on the same flash memory technology as SSDs
  - Etc

Hard disk drives



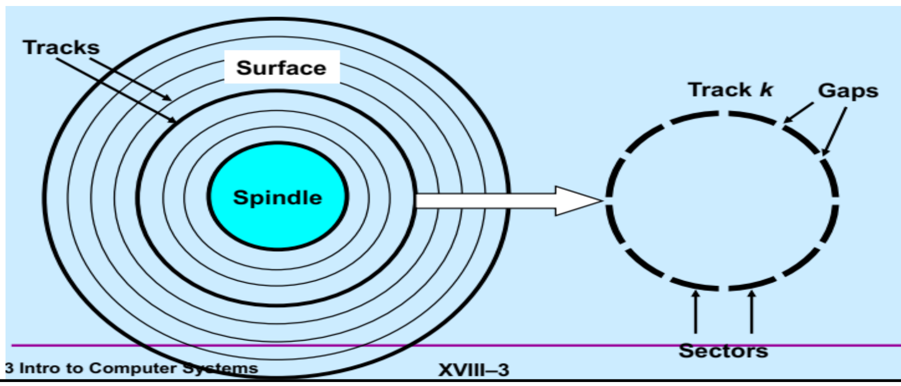
- First HDD:
  - 1956, IBM (RAMAC 305)
  - Features:
    - Weight: 1 tons
    - 50 double sided disks, 24" each
    - Two read/write heads
    - 100 tracks/disk
    - Access time: 1s
    - Capacity: 5 million 7-bit characters
- Microdrive
  - 2006: 1", 8 GB capacity

What's Inside A Disk Drive?



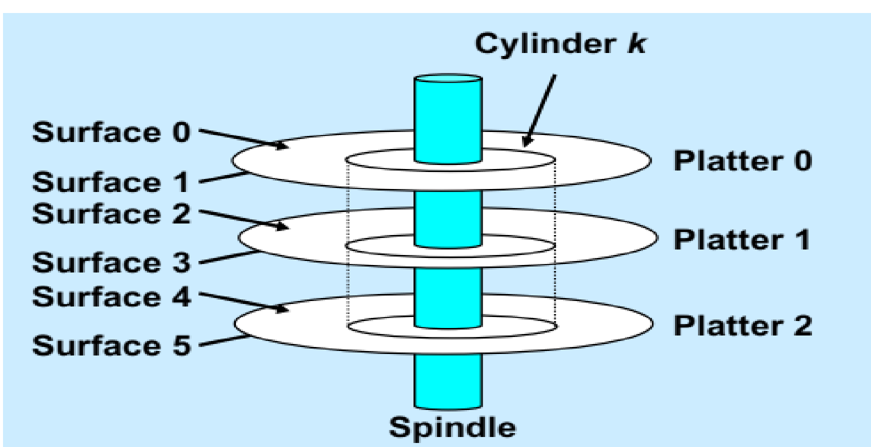
Disk geometry

- Disks consist of platters, each with two surfaces
- Each surface consists of concentric rings called tracks
- Each track consists of sectors separated by gaps



Disk geometry (Multiple – platter view)

- Aligned tracks form a cylinder



Disk capacity

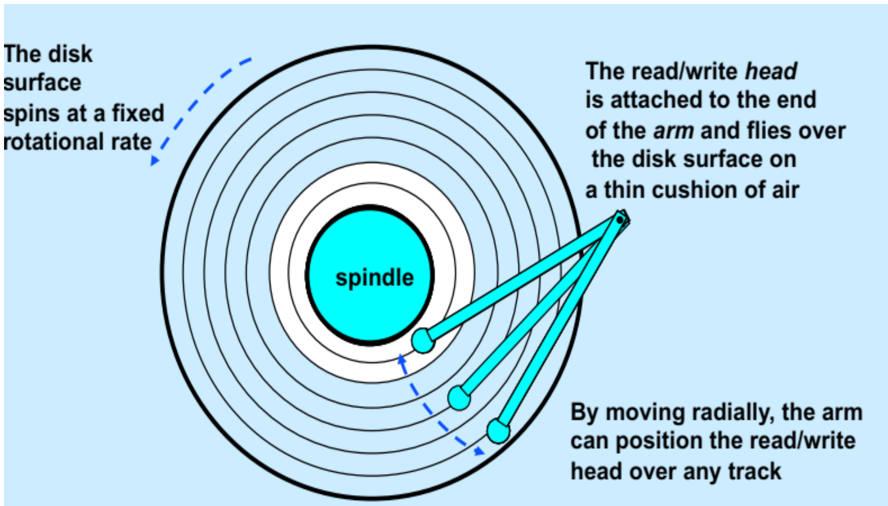
- Capacity: maximum number of bits that can be stored
  - capacity expressed in units of gigabytes (GB), where 1 GB = 2^{30} Bytes $\approx 10^9$ Bytes
- Capacity is determined by these technology factors:
  - recording density (bits/in): number of bits that can be squeezed into a 1 inch segment of a track
  - track density (tracks/in): number of tracks that can be squeezed into a 1 inch radial segment
  - areal density (bits/in2): product of recording and track density
- Modern disks partition tracks into disjoint subsets called recording zones
  - each track in a zone has the same number of sectors, determined by the circumference of innermost track
  - each zone has a different number of sectors/track

Computing disk capacity

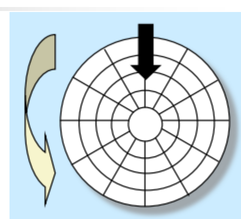
- Capacity = (# bytes/sector) x (avg. # sectors/track)
  x(# tracks/surface) x
  (#surfaces/platter) x(# platters/disk)

  Example:
  - 512 bytes/sector
  - 600 sectors/track (on average)
  - 40,000 tracks/surface
  - 2 surfaces/platter
  - 5 platters/disk
- Capacity = 512 x 600 x 40000 x 2 x 5= 122,280,000,000 = 113.88 GB

Disk operation



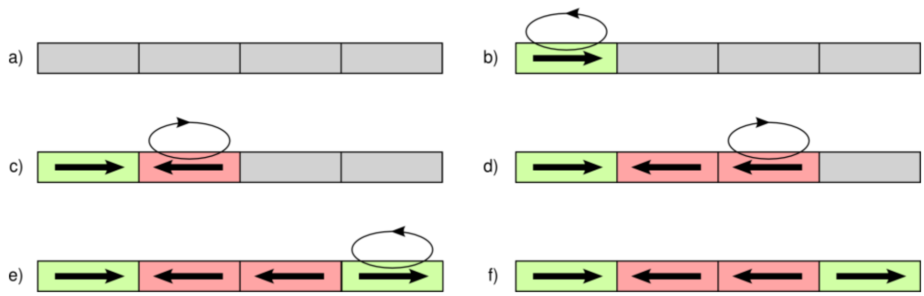
Disk structure : top view of single platter



- Surface organized into tracks
- Tracks divided into sectors
- Disk access
  - Head in position above a track
  - Head in position above a track

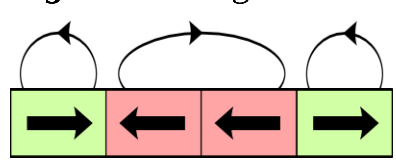
Writing data to manegtic surface

- The head is moved to the desired radial position → **seek**
- The disk is rotated to the desired angular position
- The head generates a local external magnetic field above the disk
- The disk will be magnetized permanently (locally)



Reading data from manegtic surface

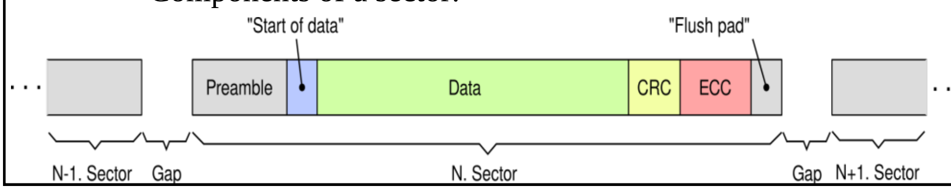
- We need to detect the magnetic field of the disk → Not possible in a direct way!
- What is possible: to detect the ***change*** of the magnetic field
  - Magnetic field is changed: bit 1
  - No change: bit 0
- Example: bit sequence „101”:



- Consequences:
  - Individual bits can not be modified!
  - Since by allowing it we would have to change the direction of the magnetic field on each subsequent bit positions
  - What we do instead: we introduce larger data units (called sectors)
  - Only whole sectors can be read or written

Data organization

- **Data units**
  - We can only read and write *blocks* (and not individual bytes)
  - **Sector system**
    - Fixed data units – sectors (typically 512 bytes)
    - Advantage: easier to handle, the free space is not fragmented
    - Issue: the operating system has to map the files of various sizes to the fixed sized sectors
    - Components of a sector:



- Gap: to leave time to switch on and off the read or write head
- Preamble: calibrate head (adjusts signal strength and data density)
- Data starts" inticates the end of calibration
- Flush pad" to time the last bytes leave the head

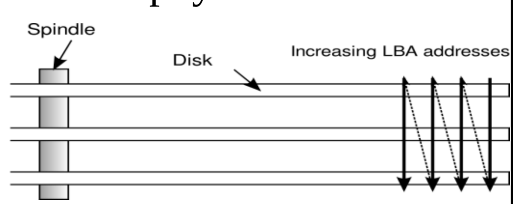
Identifying a sector

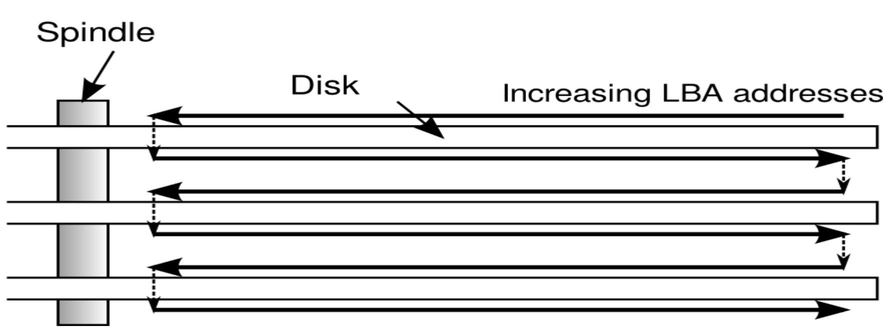
- How to refer to a sector?
- Specifying the physical position:
  - *Track*: the radial position of the data
  - Specifying tracks:
    - *Cylinder*: the same tracks of the all the platters
    - *Head*: which platter on the same cylinder
  - Specifying the locations of a sector:
    → **CHS coordinates (cylinder-head-sector)**
    (On which cylinder, under which head, which sector)
- This is how the HDD identifies a sector internally

- And how does the external environment of the HDD identify a sector?
- When the operating system wants to load a sector, how does it refer to it?
  → **By using logical addresses**
- **Logical addresses**
  - Why? Why does not the operating system use CHS?
  - They used it in the old days. Issues:
    - The HDD can not hide the bad sectors from the operating system
    - The ATA standard was able to handle 8.4 GB disks using CHS

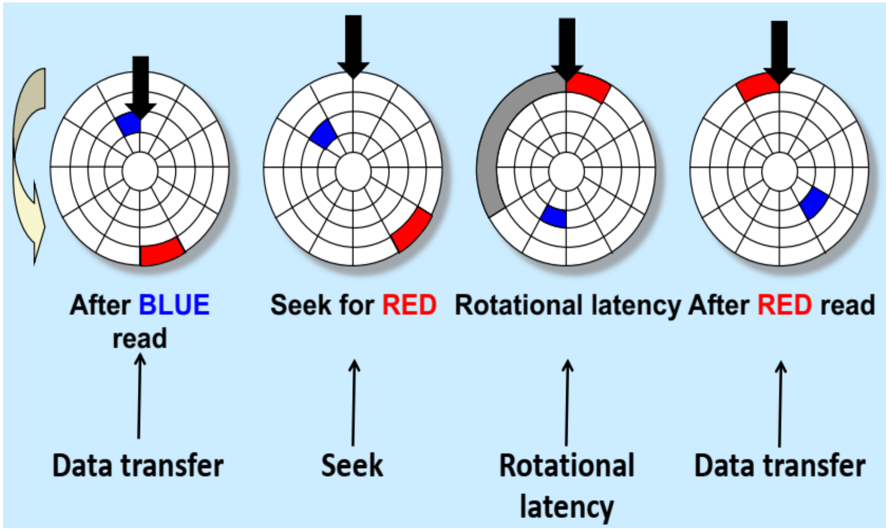
- They introduced the logical addressing: **Logical Block Address, LBA**
- Sectors are identified by a single number (which is it on the disk)
- The operating system tells just a sector number to the HDD
- The HDD maps this logical address to a physical CHS address
- The HDD is a black box now!
  - The operating system does not need to know the internal structure of the disk (number of heads, number of cylinders, etc.)
  - The HDD can hide the bad sectors by its own (it leaves them out from the logical→physical mapping)

- Mapping logical addresses to physical CHS addresses
  - Cylinder" strategy:
  - Serpentine" strategy:





Disk access – service time components



Disk access time

- Average time to access some target sector approximated by :
- Taccess = Tavg seek + Tavg rotation + Tavg transfer
  - Seek time (Tavg seek)
    - time to position heads over cylinder containing target sector
    - typical Tavg seek is 3–9 ms
  - Rotational latency (Tavg rotation)
    - time waiting for first bit of target sector to pass under r/w head
    - typical rotation speed R = 7200 RPM
    - Tavg rotation = 1/2 x 1/R x 60 sec/1 min
  - Transfer time (Tavg transfer)
    - time to read the bits in the target sector
    - Tavg transfer = 1/R x 1/(avg # sectors/track) x 60 secs/1 min

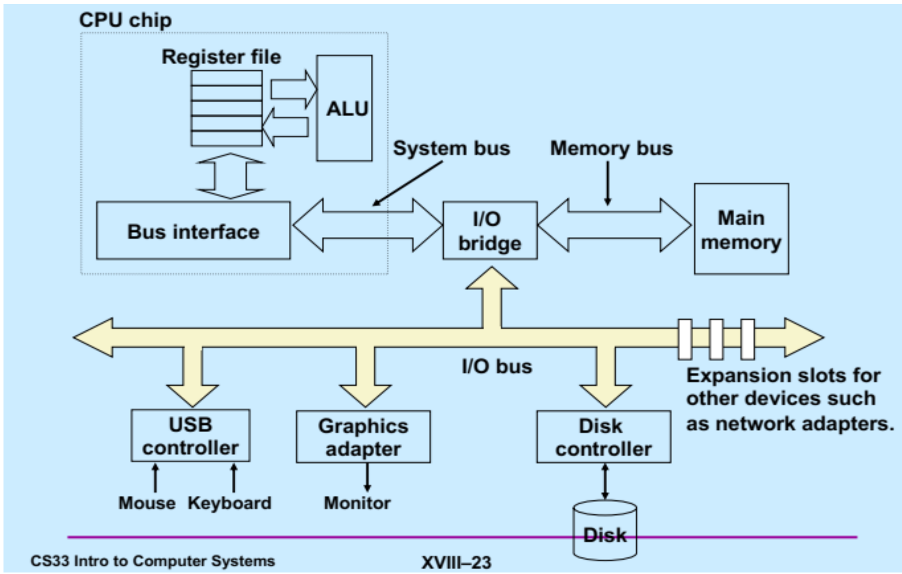
Example

- Given:
  - rotational rate = 7,200 RPM
  - average seek time = 9 ms
  - avg # sectors/track = 600
- Derived:
  - Tavg rotation = 1/2 x (60 secs/7200 RPM) x 1000 ms/sec = 4 ms
  - Tavg transfer = 60/7200 RPM x 1/600 sects/track x 1000 ms/sec = 0.014 ms
  - Taccess = 9 ms + 4 ms + 0.014 ms
- Important points:
  - access time dominated by seek time and rotational latency
  - first bit in a sector is the most expensive, the rest are free
  - SRAM access time is about 4 ns/doubleword, DRAM about 60 ns
    - disk is about 40,000 times slower than SRAM
    - 2,500 times slower than DRAM

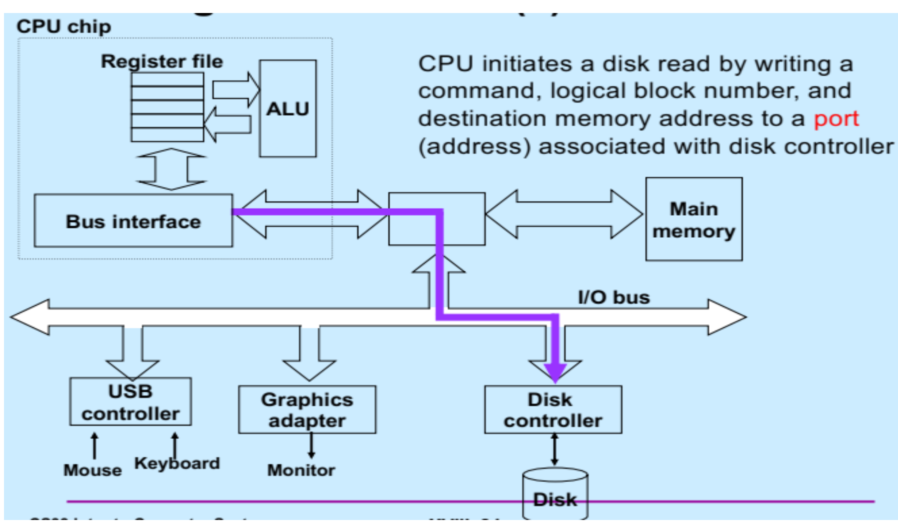
Logical disk blocks

- Modern disks present a simpler abstract view of the complex sector geometry:
  - the set of available sectors is modeled as a sequence of b-sized logical blocks (0, 1, 2, ...)
- Mapping between logical blocks and actual (physical) sectors
  - maintained by hardware/firmware device called disk controller
  - converts requests for logical blocks into (surface, track, sector) triples
- Allows controller to set aside spare cylinders for each zone
  - accounts for the difference in “formatted capacity” and

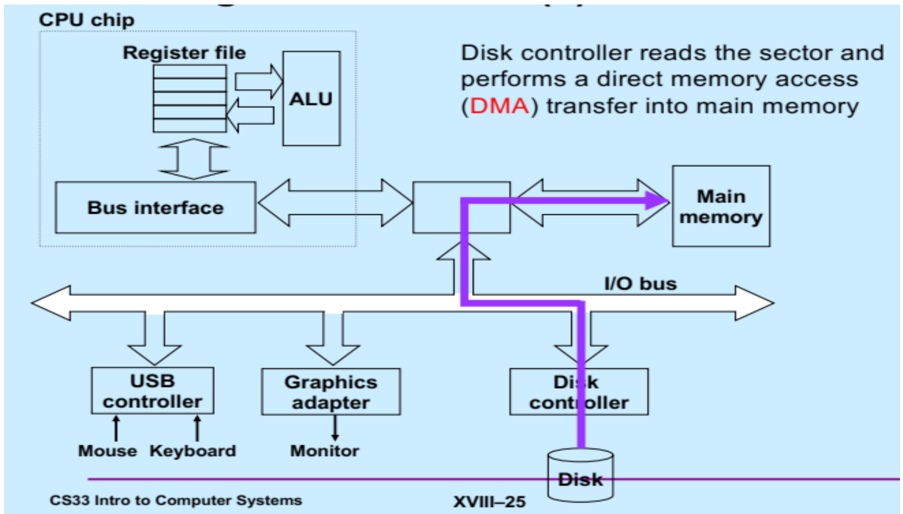
IO Bus



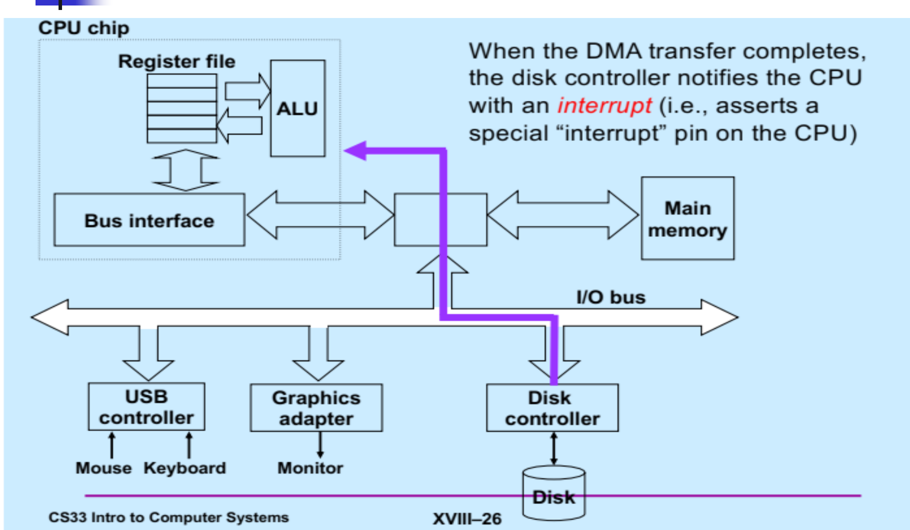
Reading a disk sector - 1



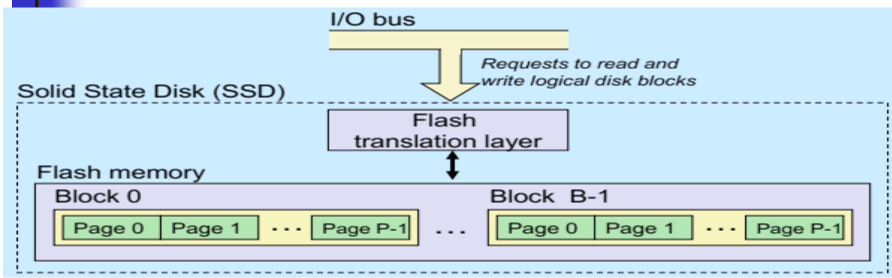
Reading a disk sector - 2



Reading a disk sector - 3



Solid – State Disks (SSDs)



- Pages: 512KB to 4KB; blocks: 32 to 128 pages
- Data read/written in units of pages
- Page can be written only after its block has been erased
- A block wears out after 100,000 repeated writes

SSD Performance Characteristics

Sequential read tput	250 MB/s	Sequential write tput	170 MB/s
Random read tput	140 MB/s	Random write tput	14 MB/s
Random read access	30 us	Random write access	300 us

- Why are random writes so slow?
  - erasing a block is slow (around 1 ms)
  - modifying a page triggers a copy of all useful pages in the block
    - find a used block (new block) and erase it
    - write the page into the new block
    - copy other pages from old block to the new block

SSD Tradeoffs vs Rotating Disks

- Advantages
  - no moving parts à faster, less power, more rugged
- Disadvantages
  - have the potential to wear out
    - mitigated by "wear-leveling logic" in flash translation layer
    - e.g. Intel X25 guarantees 1 petabyte (1015 bytes) of random writes before they wear out
  - in 2010, about 100 times more expensive per byte
  - in 2017, about 6 times more expensive per byte
- Applications
  - smart phones, laptops
  - Apple "Fusion" drives

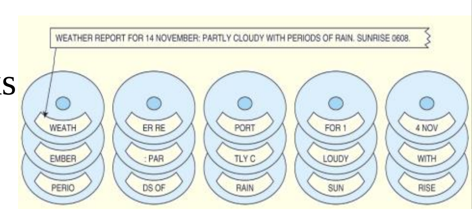
RAID

- Redundant Array of Independent Disks
- Redundant Array of Inexpensive Disks
- 6 levels in common use
- Not a hierarchy
- Set of physical disks viewed as single logical drive by O/S
- Data distributed across physical drives
- Can use redundant capacity to store parity information

- RAID, an acronym for *Redundant Array of Independent Disks* was invented to address problems of disk reliability, cost, and performance.
- In RAID, data is stored across many disks, with extra disks added to the array to provide error correction (redundancy).
- The inventors of RAID, David Patterson, Garth Gibson, and Randy Katz, provided a RAID taxonomy that has persisted for a quarter of a century, despite many efforts to redefine it.

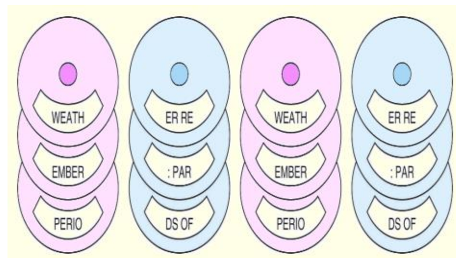
RAID 0

- RAID Level 0, also known as *drive spanning*, provides improved performance, but no redundancy.
  - Data is written in blocks across the entire array
  - The disadvantage of RAID 0 is in its low reliability.
- No redundancy
- Data striped across all disks
- Round Robin striping
- Increase speed
  - Multiple data requests probably not on same disk
  - Disks seek in parallel
  - A set of data is likely to be striped across multiple disks



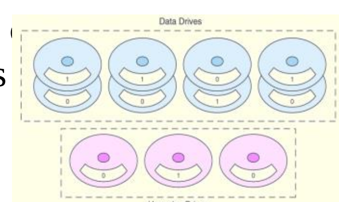
RAID 1

- Mirrored Disks, provides 100% redundancy, and good performance.
- Data is striped across disks
- 2 copies of each stripe on separate disks
- Read from either
- Write to both
- Recovery is simple
  - Swap faulty disk & re-mirror
  - No down time
- Expensive
  - Two matched sets of disks contain the same data.
  - The disadvantage of RAID 1 is cost.



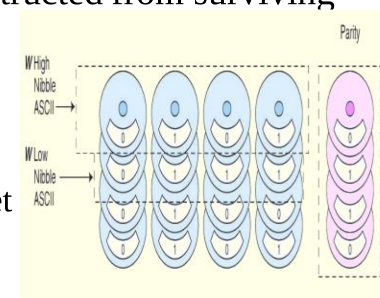
RAID 2

- Disks are synchronized
- Very small stripes
  - Often single byte/word
- Error correction calculated across corresponding bits on disks
- Multiple parity disks store Hamming code error correction in corresponding positions
- Lots of redundancy
  - Expensive
  - Not used
- A RAID Level 2 configuration consists of a set of data drives, and a set of Hamming code drives.
  - Hamming code drives provide error correction for the data drives.
  - RAID 2 performance is poor (slow) and the cost is relatively high.



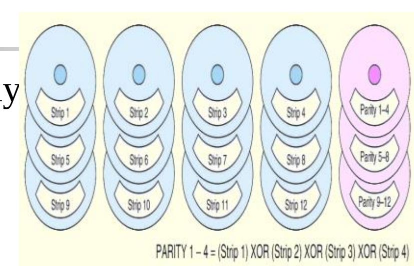
RAID 3

- Similar to RAID 2
- Only one redundant disk, no matter how large the array
- Simple parity bit for each set of corresponding bits
- Data on failed drive can be reconstructed from surviving data and parity info
- Very high transfer rates



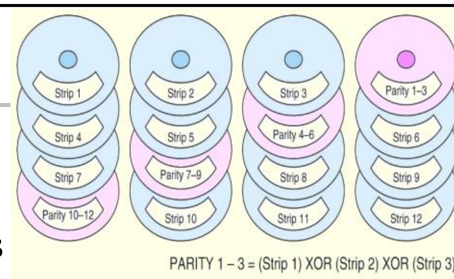
- RAID Level 3 stripes bits across a set provides a separate disk for parity.
  - Parity is the XOR of the data bits.
  - RAID 3 is not suitable for commercial applications, but is good for personal systems.

RAID 4



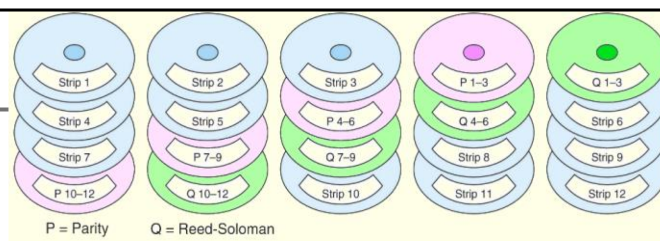
- Each disk operates independently
- Good for high I/O request rate
- Large stripes
- Bit by bit parity calculated across stripes on each disk
- Parity stored on parity disk
- RAID Level 4 is like adding parity disks to RAID 0.
  - Data is written in blocks across the data disks, and a parity block is written to the redundant drive.
  - RAID 4 would be feasible if all record blocks were the same size, such as audio/video data.
  - Poor performance, no commercial implementation of RAID

RAID 5



- Like RAID 4
- Parity striped across all disks
- Round robin allocation for parity stripe
- Avoids RAID 4 bottleneck at parity disk
- Commonly used in network servers
- N.B. DOES NOT MEAN 5 DISKS!!!!!
- RAID Level 5 is RAID 4 with distributed parity.
  - With distributed parity, some accesses can be serviced concurrently, giving good performance and high reliability.
  - RAID 5 is used in many commercial systems.

RAID 6



- Two parity calculations
- Stored in separate blocks on different disks
- User requirement of N disks needs N+2
- High data availability
  - Three disks need to fail for data loss
  - Significant write penalty
- RAID Level 6 carries two levels of error protection over striped data: Reed-Soloman and parity.
  - It can tolerate the loss of two disks.
  - RAID 6 is write-intensive, but highly fault-tolerant.

Optical Disks

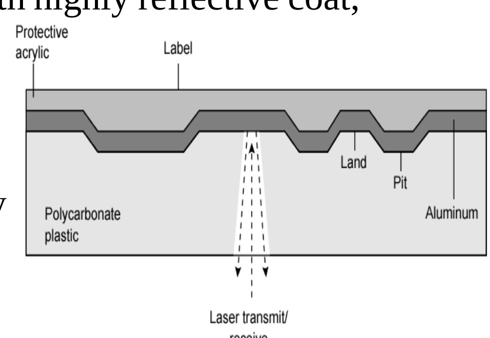
- Optical disks provide large storage capacities very inexpensively.
- They come in a number of varieties including CD-ROM, DVD, and WORM (write-once-read-many-times).
- Many large computer installations produce document output on optical disk rather than on paper. This idea is called COLD-- *Computer Output Laser Disk*.
- It is estimated that optical disks can endure for a hundred years. Other media are good for only a decade-- at best.

- CD-ROMs were designed by the music industry in the 1980s, and later adapted to data.
- This history is reflected by the fact that data is recorded in a single spiral track, starting from the center of the disk and spanning outward.
- Binary ones and zeros are delineated by bumps in the polycarbonate disk substrate. The transitions between pits and lands define binary ones.
- If you could unravel a full CD-ROM track, it would be nearly five miles long!

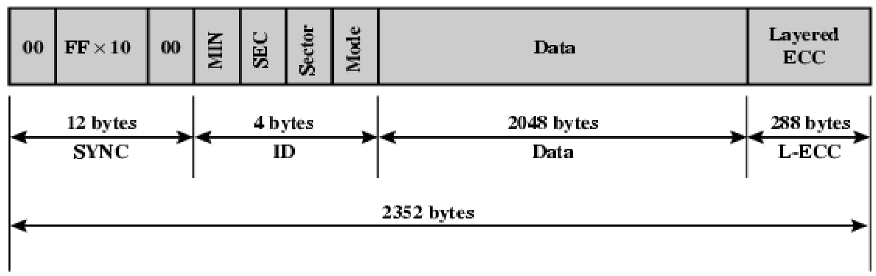
- The logical data format for a CD-ROM is much more complex than that of a magnetic disk. (See the text for details.)
- Different formats are provided for data and music.
- Two levels of error correction are provided for the data format.
- DVDs can be thought of as quad-density CDs.
- Where a CD-ROM can hold at most 650MB of data, DVDs can hold as much as 8.54GB.
- It is possible that someday DVDs will make CDs obsolete.

Optical Storage CD-ROM

- Originally for audio
- 650Mbytes giving over 70 minutes audio
- Polycarbonate coated with highly reflective coat, usually aluminium
- Data stored as pits
- Read by reflecting laser
- Constant packing density
- Constant linear velocity



CD-ROM Format



- Mode 0=blank data field
- Mode 1=2048 byte data+error correction
- Mode 2=2336 byte data