Secondary Storage Devices:
Magnetic Disks
Optical Disks
Floppy Disks
Magnetic Tapes

Secondary Storage Devices

- Two major types of secondary storage devices:

2. Direct Access Storage Devices (DASDs)
   - Magnetic Discs
     Hard disks (high capacity, low cost, fast)
     Floppy disks (low capacity, lower cost, slow)
   - Optical Disks
     CD-ROM = (Compact disc, read-only memory
3. Serial Devices
   - Magnetic tapes (very fast sequential access)

Storage and Files

- Storage has major implications for DBMS design!
  - READ: transfer data from disk to main memory (RAM).
  - WRITE: transfer data from RAM to disk.
  - Both operations are high-cost operations, relative to in-memory operations, so DB must be planned carefully!
- Why Not Store Everything in Main Memory?
  - Costs too much: Cost of RAM about 100 times the cost of the same amount of disk space, so relatively small size.
  - Main memory is volatile.
  - Typical storage hierarchy:
    - Main memory (RAM) (primary storage) for currently used data.
    - Disk for the main database (secondary storage).
    - Tapes and disks for archiving older versions of the data (tertiary storage).

Storage Hierarchy

- **Primary storage** : random access memory (RAM)
  - typical capacity a number of GB
  - cost per MB $2-3.00
  - typical access time 5ns to 60ns
- **Secondary storage**: magnetic disk/ optical devices/ tape systems
  - typical capacity a number of 100GB for fixed media; ∞ for removable
  - cost per MB $0.01 for fixed media, a little more for removable
  - typical access time 8ms to 12ms for fixed media, larger for removable

Units of Measurement

Spatial units:

- byte: 8 bits
- kilobyte (KB): 1024 or 2^{10} bytes
- megabyte (MB): 1024 kilobytes or 2^{20} bytes
- gigabyte (GB): 1024 megabytes or 2^{30} bytes

Time units:

- nanosecond (ns) one- billionth (10^{-9}) of a second
- microsecond (μ s) one- millionth (10^{-6}) of a second
- millisecond (ms) one- thousandth (10^{-3}) of a second

Primary versus Secondary Storage

– Primary storage costs several hundred times as much per unit as secondary storage, but has access times that are 250,000 to 1,000,000 times faster than secondary storage.

Memory Hierarchy

- At the primary storage level, the memory hierarchy includes, at the most expensive end' cache memory, which is a static RAM (Random Access Memory).
- The next level of primary storage is DRAM (Dynamic RAM), The advantage of DRAM is its low cost, lower speed compared with static RAM.
- Programs normally reside and execute in DRAM.
- Now that personal computers and workstations have tens of gigabytes of data in DRAM, in some cases, entire databases can be kept in the main memory (with a backup copy on magnetic disk), leading to main memory databases.

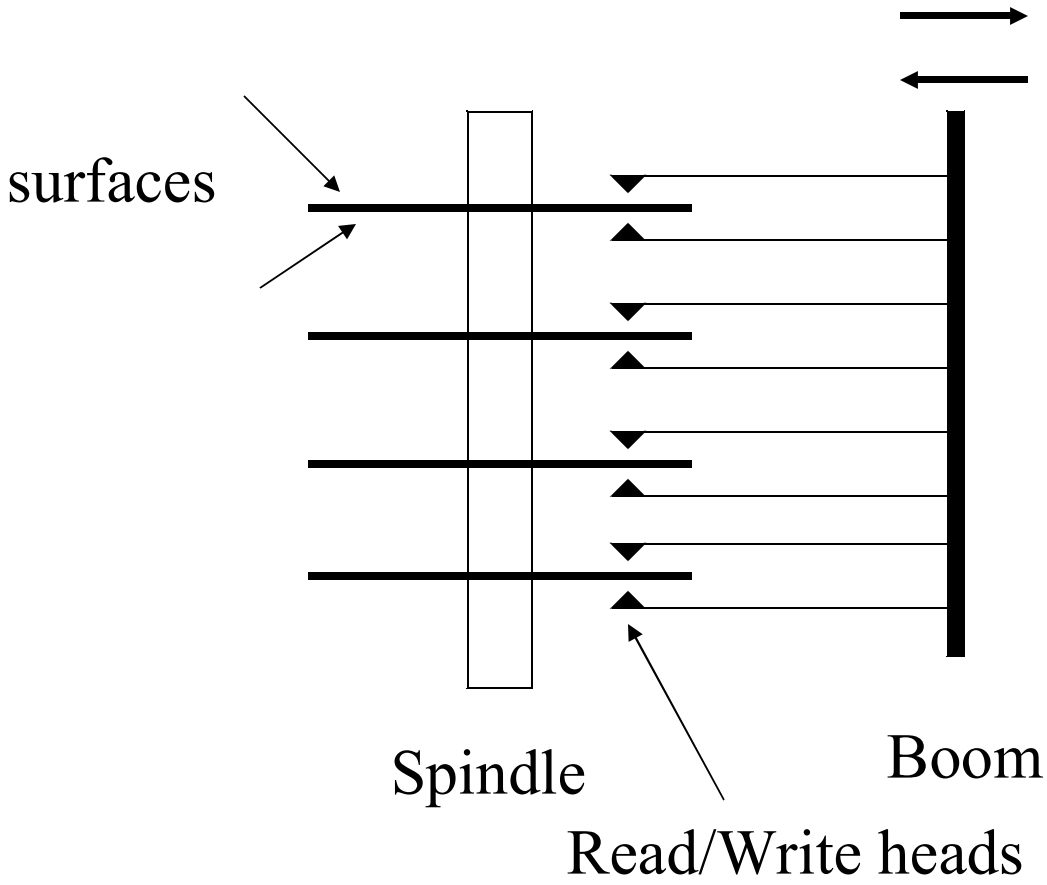
Memory Hierarchy-flash memory

- Flash memory, since 1988 it has become common, particularly because it is nonvolatile, using EEPROM (Electrically Erasable Programmable Read-Only Memory) technology. Its life is 10,000-1,000,000 times erase… Read/write is fast, but erase is slow…
- Therefore special arrangements are made for the file system, regarding file delete or update.
- Capacities up to 128 GB has been realized todate.

Magnetic Disks

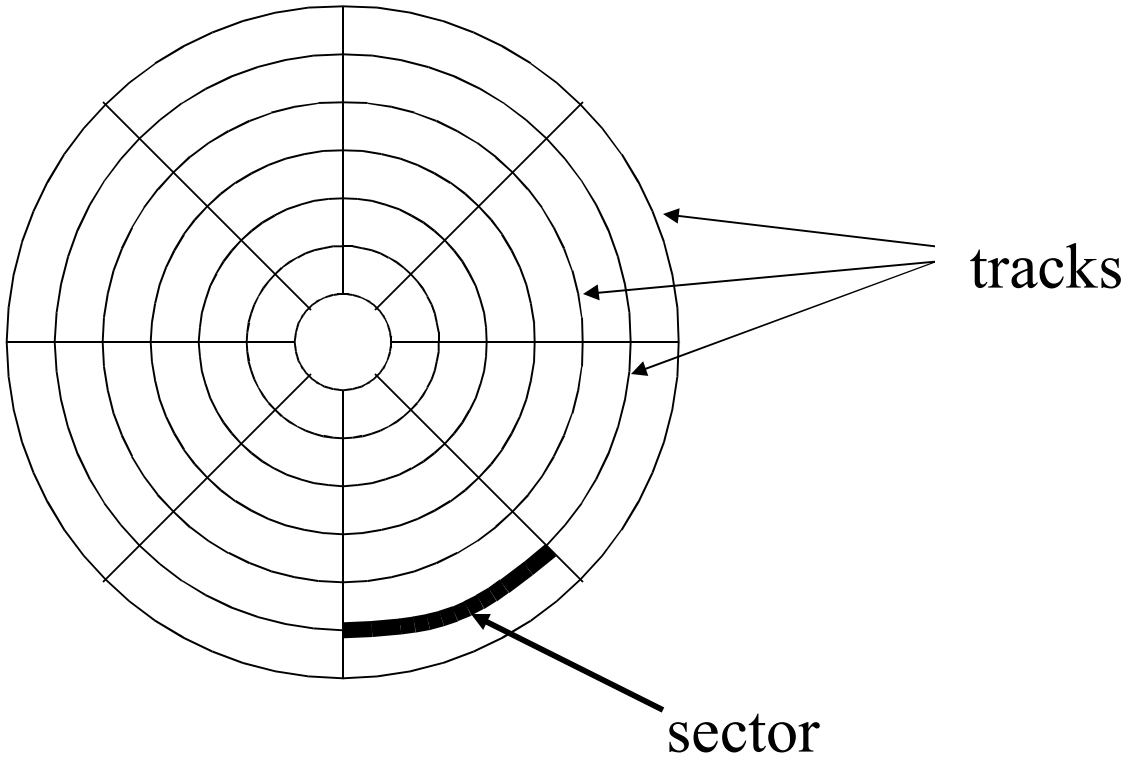
- Bits of data (0's and 1's) are stored on circular magnetic platters called disks.
- A disk rotates rapidly (& never stops).
- A disk head reads and writes bits of data as they pass under the head.
- Often, several platters are organized into a disk pack (or disk drive).

A Disk Drive



Disk drive with 4 platters and 8 surfaces and 8 RW heads

Looking at a surface



Surface of disk showing tracks and sectors

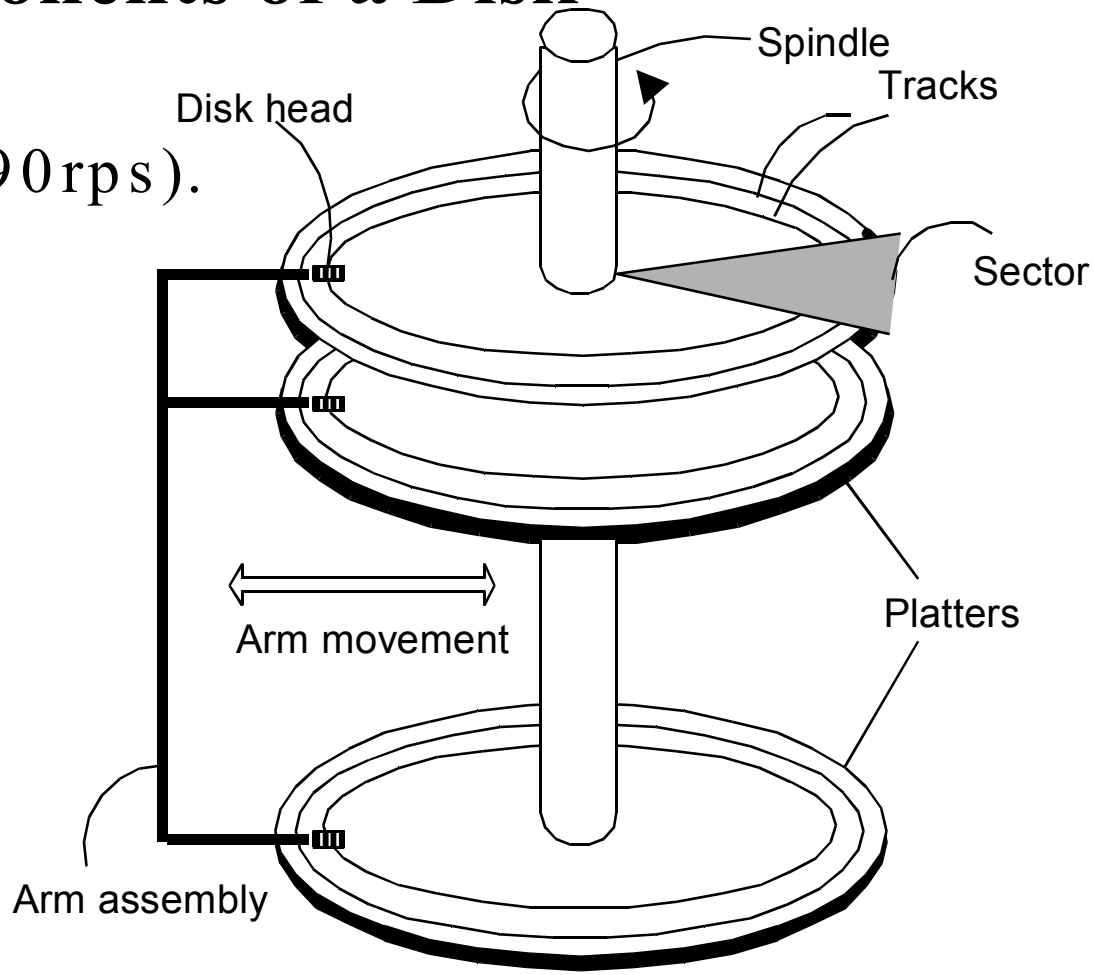
Organization of Disks

- Disk contains concentric **tracks.**
- Tracks are divided into **sectors**
- A **sector** is the smallest addressable unit in a disk.

Components of a Disk

The platters spin (say, 90rps).

- The arm assembly is moved in or out to position a head on a desired track. Tracks under heads make a cylinder (imaginary!).
- Only one head reads/writes at any one time.
- Block size is a multiple of sector size (which is often fixed).



Disk Controller

- Disk controllers: typically embedded in the disk drive, which acts as an interface between the CPU and the disk hardware.

- The controller has an internal cache (typically a number of MBs) that it uses to buffer data for read/write requests.

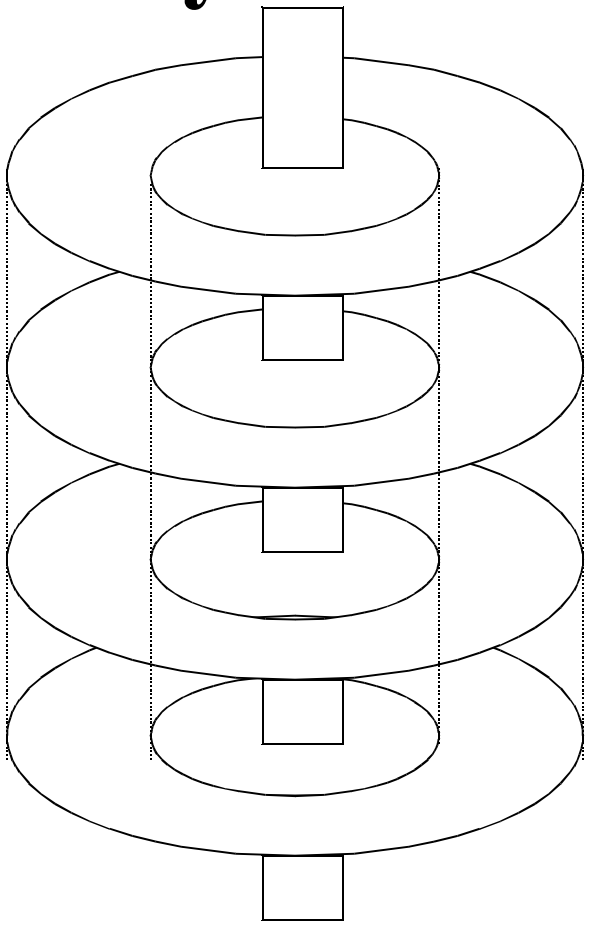
Accessing Data

- When a program reads a byte from the disk, the operating system locates the surface, track and sector containing that byte, and reads the entire sector into a special area in main memory called **buffer**.
- The bottleneck of a disk access is moving the read/ write arm.
  - So it makes sense to store a file in tracks that are below/above each other on different surfaces, rather than in several tracks on the same surface.

Cylinders

- A **cylinder** is the set of tracks at a given radius of a disk pack.
  - i.e. a cylinder is the set of tracks that can be accessed without moving the disk arm.
- All the information on a cylinder can be accessed without moving the read/write arm.

Cylinders



Estimating Capacities

- Track capacity = # of sectors/track * bytes/sector
- Cylinder capacity = # of tracks/# of cylinder * track capacity
- Drive capacity = # of cylinders * cylinder capacity
- Number of cylinders = # of tracks on a surface

Exercise

- Store a file of 20000 records on a disk with the following characteristics:

  # of bytes per sector = 512

  # of sectors per track = 40

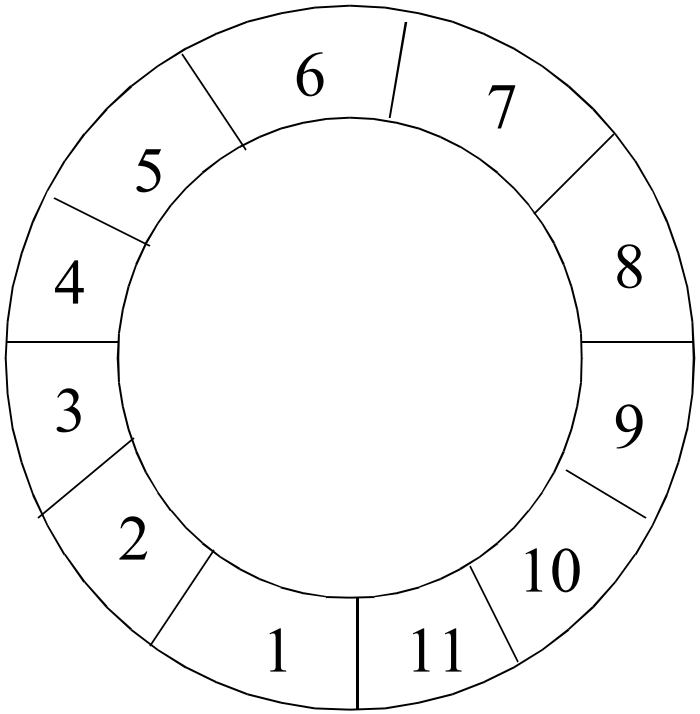
  # of tracks per cylinder = 11

  # of cylinders = 1331

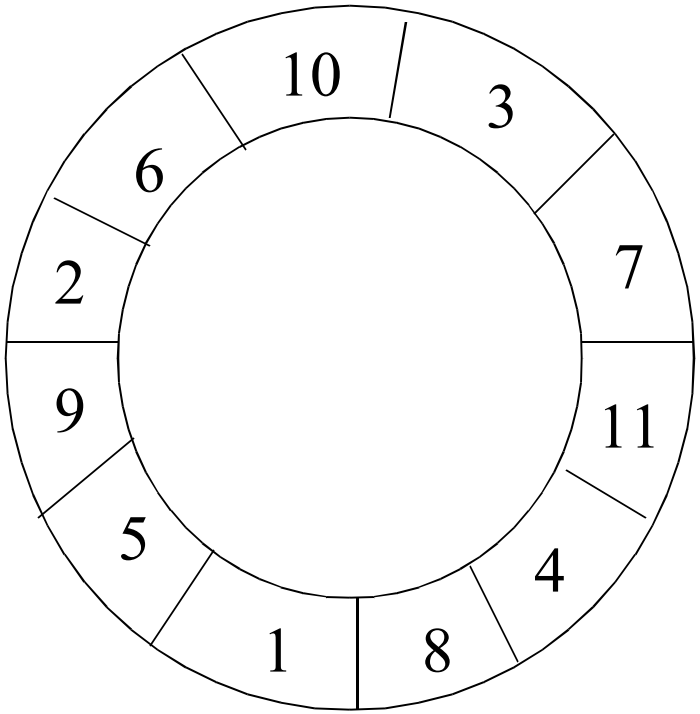
Q1. How many cylinders does the file require if each data record requires 256 bytes?

Q2. What is the total capacity of the above disk?

Organizing Tracks by sector



Physically adjacent sectors



Sectors with 3:1 interleaving

Exercise

- Suppose we want to read consecutively the sectors of a track in order: sectors 1, 2,… 11.
- How many revolutions to read the disk?
  a) Without interleaving
  b) With 3:1 interleaving
- *Note: nowadays most disk controllers are fast enough so interleaving is not common...*

Clusters

- Usually **File manager,** under the operating system, maintains the logical view of a file.
- **File manager** views the file as a series of **clusters,** each of a number of sectors. The clusters are ordered by their logical order.
- Files can be seen in the form of logical sectors or blocks, which needs to be mapped to physical clusters.
- File manager uses a **file allocation table (FAT)** to map logical sectors of the file to the physical clusters.

Extents

- If there is a lot of room on a disk, it may be possible to make a file consist entirely of contiguous clusters. Then we say that the file is one **extent**. (very good for sequential processing)
- If there isn't enough contiguous space available to contain an entire file, the file is divided into two or more noncontiguous parts. Each part is a separate extent.

External/Internal Fragmentation

- External fragmentation can be represented by a highly fragmented disk storage with many small empty holes….
- Internal fragmentation: loss of space within a sector or a cluster.
  - Due to records not fitting exactly in a sector:
    e.g. Sector size is 512 and record size is 300 bytes. Either
    - store one record per sector, or
    - allow records *span* sectors…
  - Due to the use of clusters: If the file size is not a multiple of the cluster size, then the last cluster will be partially used.

Choice of cluster size

- Some operating systems allow system administrator to choose cluster size.

- When to use large cluster size?

- What about small cluster size?

Organizing Tracks by Block

- Disk tracks may be divided into user-defined blocks rather than into sectors.
- Blocks can be fixed or variable length.
- A block is usually organized to hold an integral number of logical records.
- **Blocking Factor** = number of records stored in a block.
- No internal fragmentation, no record spanning over two blocks.
- In block-addressing scheme each block of data may be accompanied by one or more *subblocks* containing extra information about the block: record count, last record key on the block…

Non-data Overhead

- Both blocks and sectors require non-data overhead (written during formatting)
- On sector addressable disks, this information involves sector address, track address, and condition (usable/defective). Also pre-formatting involves placing gaps and synchronization marks between the sectors.
- Where a block may be of any size, more information is needed and the programmer should be aware of some of this information to utilize it for better efficiency…

Exercise

- Consider a *block-addressable* disk with the following characteristics:
  - Size of track 20,000 bytes.
  - Nondata overhead per block = 300 bytes.
  - Record size = 100 byte.
- **Q) How many records can be stored per track if blocking factor is (a) 10, (b) 60?**

  a) 10 (20000/1300*10=150)
  b) 60 (20000/6300*60=180)

The Cost of a Disk Access

- The time to access a sector in a track on a surface is divided into 3 components:

Time Component	Action
Seek Time	Time to move the read/write arm to the correct cylinder
Rotational delay (or latency)	Time it takes for the disk to rotate so that the desired sector is under the read/write head
Transfer time	Once the read/write head is positioned over the data, this is the time it takes for transferring data

Seek time

- Seek time is the time required to move the arm to the correct cylinder.
- Largest in cost.

Typically:

  - 5 ms (miliseconds) to move from one track to the next (track-to-track)
  - 50 ms maximum (from inside track to outside track)
  - 30 ms average (from one **random** track to another **random** track)

Average Seek Time (*s*)-1

- It is usually impossible to know exactly how many tracks will be traversed in every seek,
  - we usually try to determine the **average seek time** (***s***) required for a particular file operation.
- If the starting positions for each access are random, it turns out that the average seek traverses **one third of the total number of cylinders**.
  - There are more ways to travel short distance than to travel long distance…
- Manufacturer's specifications for disk drives often list this figure as the **average seek time** for the drives.
- Most hard disks today have ***s*** under 9 ms, and high-performance disks have ***s*** as low as 7.5 ms.

Average Seek Time (*s*)-2

- Seek time depends only on the speed with which the head rack moves, and the number of tracks that the head must move across to reach its target.
- Given the following (which are constant for a particular disk):
  - H_s = the time for the I/ O head to start moving
  - H_t = the time for the I/ O head to move from one track to the next
- Then the time for the head to move n tracks is:
- $Seek(n) = H_s + H_t * n$

Rotational Latency(latency)-1

- Latency is the time needed for the disk to rotate so that the sector we want is under the **read/write** head.
- Hard disks usually rotate at about 5000-7000 rpm,
  - 12-8 msec per revolution.
  - Eg. for 7200 rpm, max latency is 8.33 msec.
- Note:
  - Min latency = 0
  - Max latency = Time for one disk revolution
  - **Average latency (*r*)** = (min + max) / 2
    = max / 2
    = time for ½ disk revolution
    - Typically 6 – 4 ms, at average

Rotational Latency computation-2

- Given the following:
  - R = the rotational speed of the spindle, in number of rotations per second.
  - θ = the number of radians through which the track must rotate; that is $\theta/2\pi$ in number of rotations.
  - then the rotational latency θ radians in msec is:
- Latency= $((\theta/2\pi)/R)*1000$, *in ms*

Transfer Time-1

- Transfer time is the time for the read/write head to pass over a block.
- The transfer time is given by the formula:

$$\text{Transfer time} = \frac{\text{number of sectors to transfer}}{\text{number of sectors per track}} \times \text{rotation time}$$

  - e.g. if there are S_t sectors per track, the time to transfer one sector would be $1/ S_t$ of a revolution.

Transfer Time-2

- The transfer time depends only on the speed at which the spindle rotates, and the number of sectors that must be read.
- Given:
  - S_t = the total number of sectors per track
  - the transfer time for n contiguous sectors on the same track is:
- Transfer Time =$((n/S_t)/R)*1000$, *in ms*

Exercise

Given the following disk:

- 20 surfaces
  800 tracks/surface
  25 sectors/track
  512 bytes/sector
- 3600 rpm (revolutions per minute)
- 7 ms track-to-track seek time
  28 ms avg. seek time
  50 ms max seek time.

Find:

c) Disk capacity in bytes

d) Maximum and Average latencies

e) Total time to read the entire disk, one cylinder at a time

Exercise

- Disk characteristics:
  - Average seek time = 8 msec.
  - Average rotational latency = 3 msec
  - Maximum rotational latency = 6 msec.
  - Spindle speed = 10,000 rpm
  - Sectors per track = 25
  - Sector size = 512 bytes
- **Q) What is the average time to read one sector? Read-write head is a-on the sector, b-off the sector.**
- **a) 6/25*1000=0.24 msec, b) 0.24+8**

Sequential Reading

- Given the following disk:
  - Avg. Seek time, s = 16 ms
  - Avg. Rot. Latency, r = 8.3 ms (this is true for 3600 rpm disk)
  - Block (one sector=2400 bytes) transfer time, btt = 0.84 ms, assuming btt is equivalent to effective btt(or ebtt)

b) Calculate the time to read 10 sequential blocks, on the same track.

c) Calculate the time to read 10 sequential cylinders, if there are 200 cylinders, and 20 surfaces with 200 sectors per track.

Random Reading

Given the same disk,

b) Calculate the time to read one block randomly

c) Calculate the time to read 100 blocks randomly

d) Calculate the time to read 100 blocks sequentially.

Fast Sequential Reading-FSR

- FSR assumes that blocks are arranged so that there is no rotational delay in transferring from one track to another within the same cylinder. This is possible if consecutive track beginnings are staggered (like running races on circular race tracks)
- assumes that the consecutive blocks are arranged so that when the next block is on an adjacent cylinder, there is no rotational delay after the arm is moved to new cylinder
- So, in FSR assume no rotational delay after finding the first block.

Assuming Fast Seq. Reading

- Formulation of Reading *b* blocks:

i. Sequentially:

$$\underbrace{s + r} + b * btt$$

s+ r is insignificant for large files, where *b* is very large: thus

$$\Rightarrow \quad b * btt$$

vi. Randomly:

$$b * (s + r + btt)$$

Exercise

- Given a file of 30000 records, 1600 bytes each, and block size 2400 bytes, how does record placement affect sequential reading time, in the following cases? Discuss.
  i) Empty space in blocks-internal fragmentation.
  ii) Records overlap block boundaries.

Exercise

- **Specifications of a disk drive**:
  - Min seek time, track-to-track = 6ms.
  - Average seek time = 18ms
  - Avg. Rotational delay = 8.3ms
  - Transfer time or byte transfer rate=16.7 ms/track or 1229 bytes/ms
  - Bytes per sector = 512
  - Sectors per track = 40
  - Tracks per cylinder = 12 (number of surfaces)
  - Tracks per surface = 1331
  - Non Interleaving
  - Cluster size= 8 sectors=4096 bytes
  - Smallest extent size = 5 clusters (files can be stored in several extents…)

Q) How long will it take to read a 2048KB file that is divided into 8000 records, each record 256 bytes?

i) Access the file sequentially, ie. In physical order.

ii) Access the file randomly, in some logical record order.

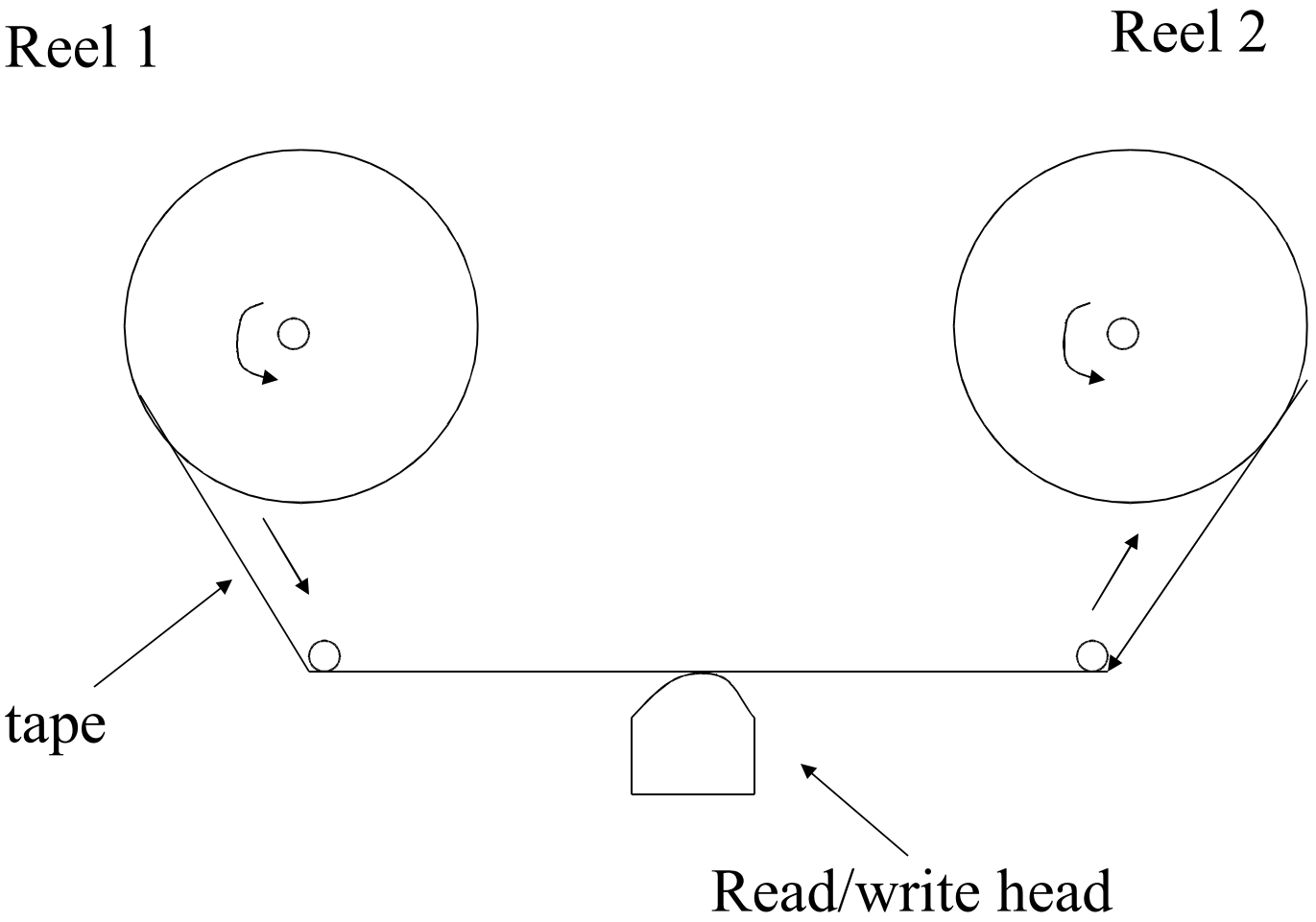
Secondary Storage Devices: Magnetic Tapes

Characteristics

- No direct access, but very fast sequential access.
- Resistant to different environmental conditions.
- Easy to transport, store, cheaper than disk.
- Before it was widely used to store application data; nowadays, it's mostly used for backups or archives.

MT Characteristics-2

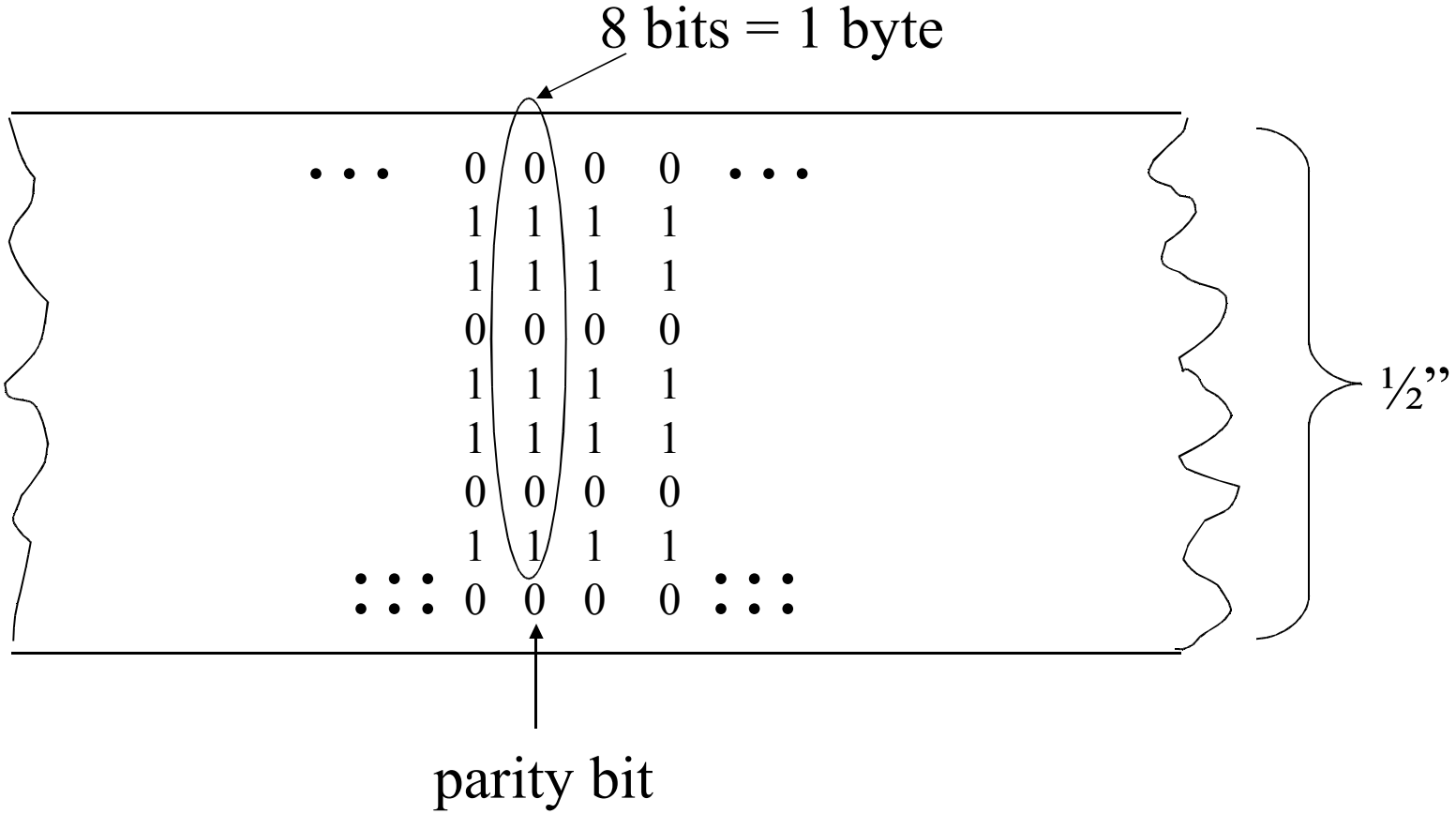
- A sequence of bits are stored on magnetic tape.
- For storage, the tape is wound on a reel.
- To access the data, the tape is unwound from one reel to another.
- As the tape passes the head, bits of data are read from or written onto the tape.


Reel 1
Reel 2
tape
Read/write head

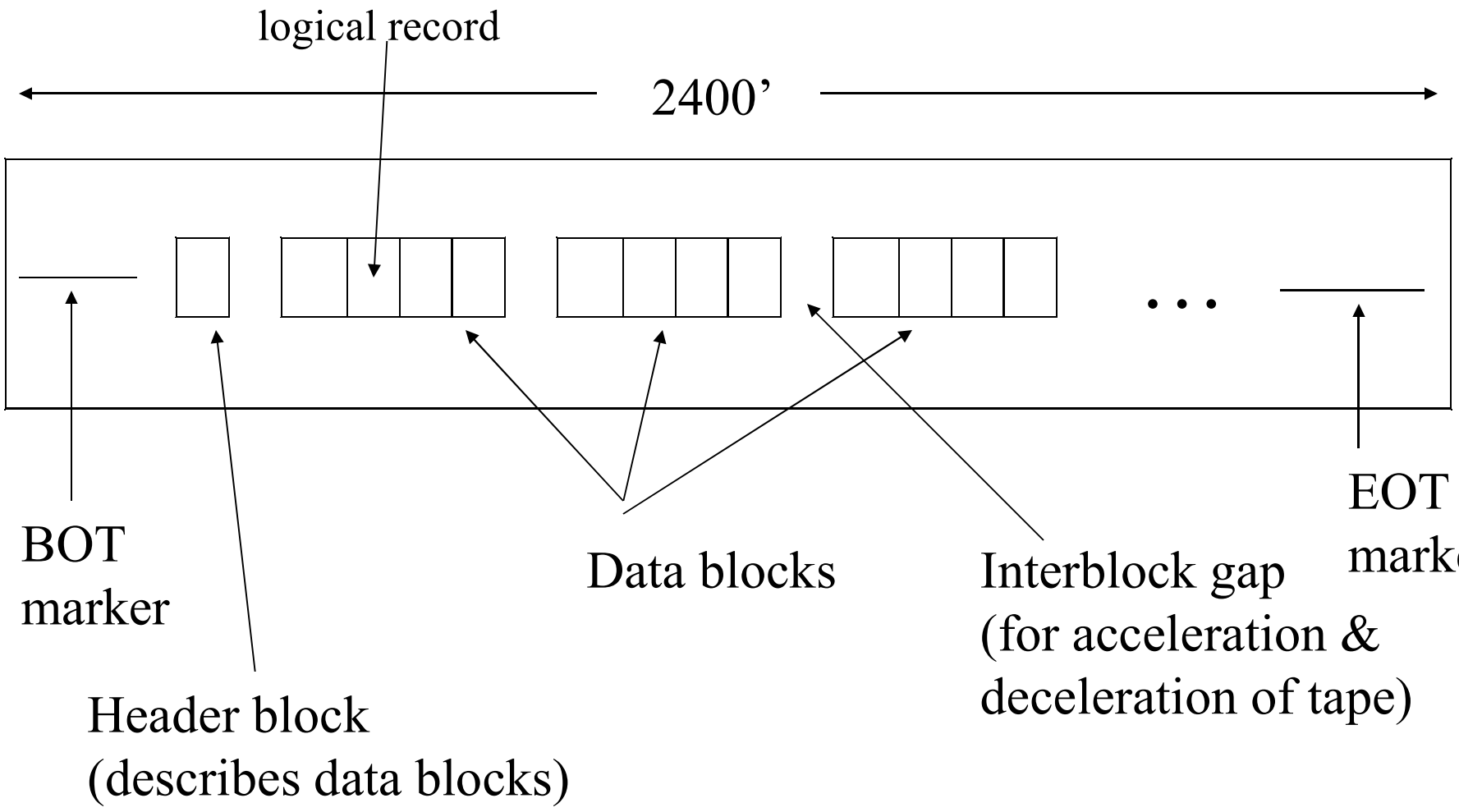
Tracks

- Typically data on tape is stored in 9 separate bit streams, or tracks.
- Each track is a sequence of bits.
- Recording density = # of bits per inch (bpi).
  Typically 800 or 1600 bpi.
  30000 bpi or more on some recent devices.

MT recording in detail



Tape Organization



Data Blocks and Records

- Each data block is a sequence of contiguous records.
- A record is the unit of data that a user's program deals with.
- The tape drive reads an entire block of records at once.
- Unlike a disk, a tape starts and stops.
- When stopped, the read/write head is over an interblock gap.

Example: tape capacity

- Given the following tape:
  - Recording density = 1600 bpi
  - Tape length = 2400 '
  - Interblockgap = ½ "
  - 512 bytes per record
  - Blocking factor = 25
- How many records can we write on the tape? (ignoring BOT and EOT markers and the header block for simplicity)

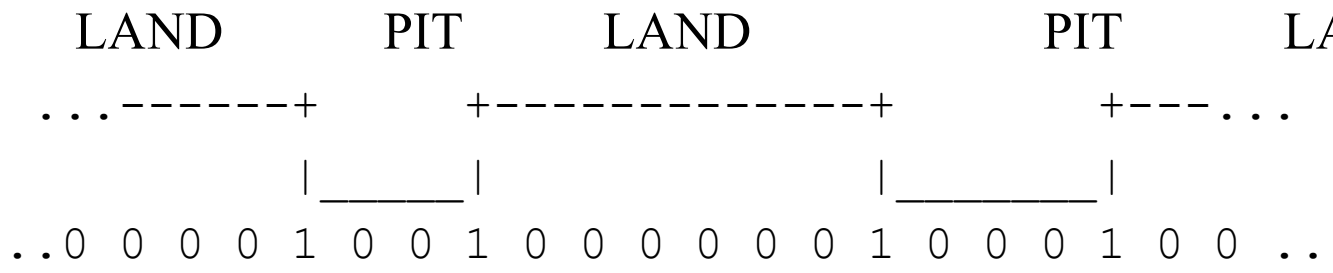
Secondary Storage Devices: CD-ROM

Physical Organization of CD-ROM

- Compact Disk – read only memory (write once), R/W is also available.
- Data is encoded and read optically with a laser
- Can store around +600MB data
- Digital data is represented as a series of **Pits** and **Lands:**
  - Pit = a little depression, forming a lower level in the track
  - Land = the flat part between pits, or the upper levels in the track

Organization of data

- Reading a CD is done by shining a laser at the disc and detecting changing reflections patterns.
  - 1 = change in height (land to pit or pit to land)
  - 0 = a “fixed” amount of time between 1’s



- Note : we cannot have two 1’s in a row!
  => uses Eight to Fourteen Modulation (EFM) encoding table. Usually, a pattern of 8 bits is translated to/from a pattern of 14 **pits** and **lands**.

CD-ROM

- While the speed of CD-ROM readers is relatively higher, such as 24X(24 times CD audio speed), the speed of writing is much slower, as low as 2X.
  - Note that the speed of the audio is about 150KB per second.
- The DVD (Digital Video Disc or Digital Versatile Disc) technology is based on CD technology with increased storage density. 1x means 1.32 MB, equivalent 9x CD-ROM speed.
- The DVD technology allows two-side medium, which may extend the CD-ROM to a storage capacity of up to 21GB, using special SW and HW decoder....

DVD vs CD-ROM

The main differences between the CD and DVD are summarized in the following table:

Feature	DVD	CD-ROM
Substrate diameter / thickness (mm)	120 / 1.2	120 / 1.2
Sides	1 or 2	1
Layers per side	1 or 2	1
Capacity (GB)	upto ~20GB	~ 0.7
Track pitch (microns)	0.74	1.6
Min pit length (microns)	0.4 - 0.44	0.83
Linear velocity used for scan (m/s)	3.5 - 3.84	1.3
Laser wavelength (nm)	635 or 650	780
Numerical aperture	0.6	0.45
Modulation	8 to 16	EFM (8 to 17)
Error correction code (ECC)	RSPC	CIRC
Durability and dust/scratch	same as that of CD	high

CD-ROM

- Because of the heritage from CD audio, the data is stored as a single spiral track on the CD-ROM, contrary to magnetic hard disk's discrete track concept.
- Thus, the rotation speed is controlled by CLV-Constant Linear velocity, Constant Angular Velocity-CAV, or Zoned Constant Linear Velocity. IN CLV case, the rotational speed at the center is highest, slowing down towards the outer edge. Because, the recording density is the same every where.
- Note that with CLV, the linear speed of the spiral passing under the R/W head remains constant.
- CLV is the culprit for the poor seek time in CD-ROMs
- The advantage of CLV is that the disk is utilized at its best capacity, as the recording density is the same every where.

CD-ROM

- Note that: Since 0's are represented by the **length of time** between transitions, we must travel at **constant linear velocity** (**CLV**)on the tracks.
- Sectors are organized along a spiral
- Sectors have same linear length
- Advantage: takes advantage of all storage space available.
- Disadvantage: has to change rotational speed when seeking (slower towards the outside)

CD-ROM

Question: Why does it take only 70 minutes of playing time in an CD audio.

- Ans.: If the sound frequency is 20 kilohertz, we need twice as much frequency for sampling speed to reconstruct the sound wave. Each sample may take up to 2 bytes.
  - An accepted standard allows a sampling speed of 44100 times per second, which requires 88200 bytes for 2 bytes per sample. For stereo, this becomes 176400 bytes per second.
  - If the capacity is about 600MB, you can compute number of minutes required…

Addressing

- 1 second of play time is divided up into 75 **sectors**.
- Each sector holds 2KB
- 60 min CD:
  60min * 60 sec/min * 75 sectors/sec =
  270,000 sectors = 540,000 KB ~ 540 MB
- A **sector** is addressed by:
  Minute:Second:Sector
  e.g. 16:22:34

File Structures for CD-ROM

- One of the problems faced in using CDs for data storage is acceptance of a common file system, with the following desired design goals:
  - Support for hierarchical directory structure, with access of one or two seeks…
  - Support for generic file names (as in “file*.c”), during directory access
- If implement UNIX file system on CD-ROM, it will be a catastrophe! The seek time per access is from 500 msec to 1 sec.

File Structures for CD-ROM

- In this case, one seek may be necessary per subdirectory. For example /usr/home/mydir/ceng351/exam1

  will require five seeks to locate the file exam1 only…
- Solution
  - One approach place the entire directory structure in one file, such that it allows building a left child right sibling structure to be able to access any file.
- For a small file structure file, the entire directory structure can be kept in the memory all the time, which allows method to work.

File Structures for CD-ROM

- The second approach is to create an index to the file locations by hashing the full path names of each file.
- This method will not work for generic file or directory searches.
- A third method may utilize both above methods, one can keep the advantage of Unix like one file per directory scheme, at the same time allows building indexes for the subdirectories.

File Structures for CD-ROM

- A forth method, assume directories as files as well and use a special index that organizes the directories and the files into a hierarchy where a simple parental index indicates the relationship between all entries.

Rec Number	File or dir name	Parent
0	Root	
1	Subdir1	0
2	Subdir11	1
3	Subdir12	1
4	File11	1
5	File	0
6	Subdir2	0

Representation of individual files on CD-ROM

- B+ Tree type data structures are appropriate for organizing the files on CD-ROMs.
- **Build once read many times** allows attempting to achieve100% utilization of blocks or buckets. Packing the internal nodes so that all of them can be maintained in the memory during the data fetches is important.
- Secondary indexes can be formed so that the records are pinned to the indexes on a CD-ROM, as the file will never be reorganized…

Representation of individual files on CD-ROM

- This may force the files on the source disks and their copies on the CD-ROM to be differently organized, because of the efficiency concerns.
- It is possible to use hashing on the CD-ROM, except that the overflow should either not exist or minimized. This becomes possible when the addressing space is kept large.
- Remember that the files to be put on a CD-ROM are final, so the hashing function can be chosen to perform the best, i.e. with no collisions.

A journey of a Byte
and
Buffer Management

A journey of a byte

- Suppose in our program we wrote:

```
outfile << c;
```

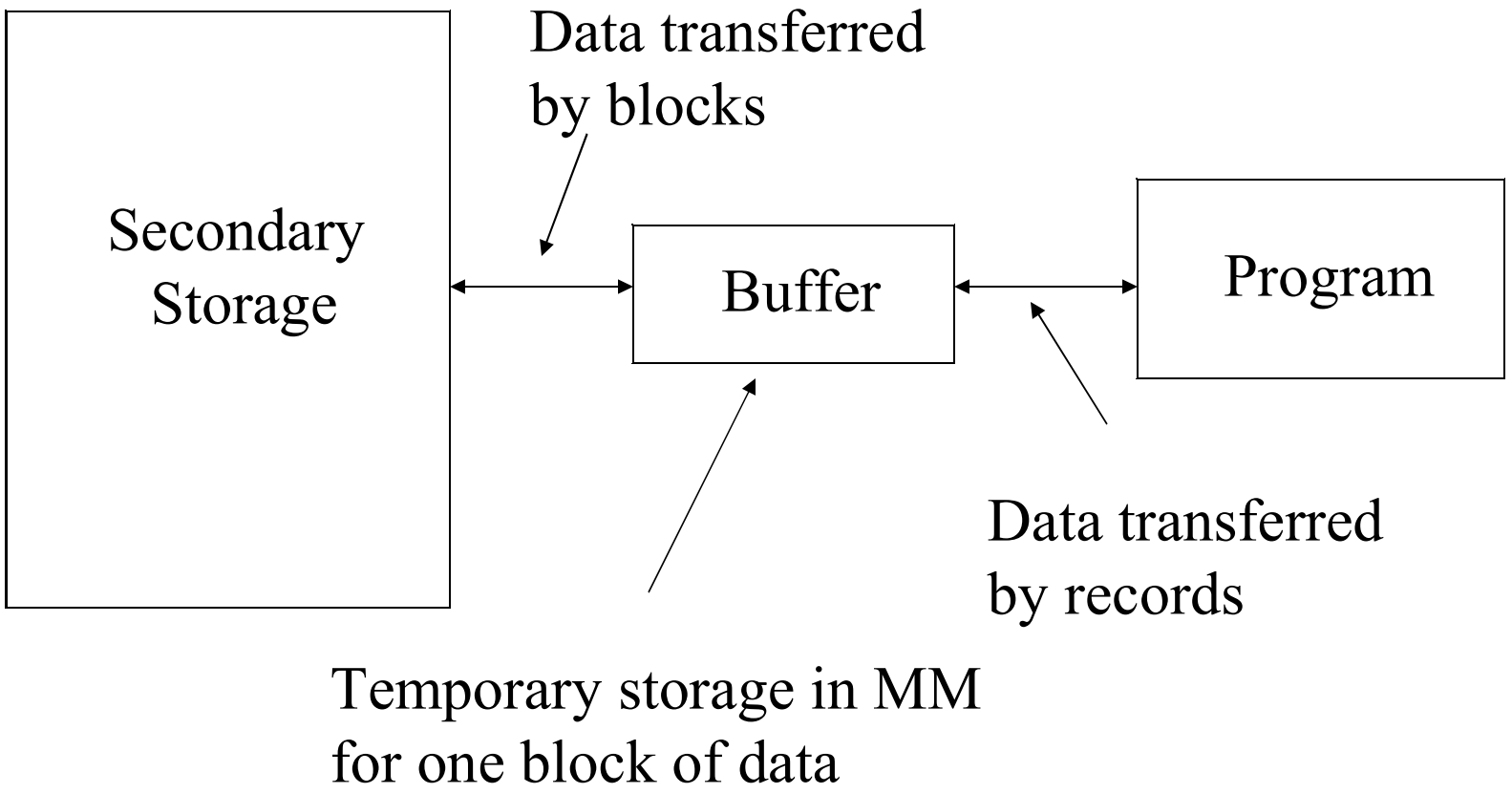
- This causes a call to the **file manager** (a part of O.S. responsible for I/O operations)
- The O/S (File manager) makes sure that the byte is written to the disk.
- Pieces of software/hardware involved in I/O:
  - Application Program
  - Operating System/ file manager
  - I/O Processor
  - Disk Controller

- **Application program**
  - Requests the I/O operation
- **Operating system / file manager**
  - Keeps tables for all opened files
  - Brings appropriate sector to buffer.
  - Writes byte to buffer
  - Gives instruction to I/O processor to write data from this buffer into correct place in disk or vice versa.
  - Note: the buffer is an exact image of a cluster in disk.
- **I/O Processor**
  - a separate chip; runs independently of CPU
  - Find a time when drive is available to receive data and put data in proper format for the disk
  - Sends data to disk controller
- **Disk controller**
  - A separate chip; instructs the drive to move R/W head
  - Sends the byte to the surface when the proper sector comes under R/W head.

Buffer Management

- Buffering means working with large chunks of data in the main memory so the number of accesses to secondary storage is reduced.
- System I/O buffers are beyond the control of application programs and are manipulated by the OS.
- Note that the application program may implement its own “buffer” – i.e. a place in memory (variable, object) that accumulates large chunks of data to be later written to disk as one chunk.

System I/O Buffer



Buffer Bottlenecks

- Consider the following program segment:

```
while (1) {
  infile >> ch;
    if (infile.fail()) break;
    outfile << ch;
}
```

- What happens if the OS used only one I/O buffer?

  ⇒ Buffer bottleneck

- Most OS have an input buffer and an output buffer.

Buffering Strategies

- **Double Buffering:** Two buffers can be used to allow processing and I/O to overlap.
  - Suppose that a program is only writing to a disk.
  - CPU wants to fill a buffer at the same time that I/O is being performed.
  - If two buffers are used and I/O-CPU overlapping is permitted, CPU can be filling one buffer while the other buffer is being transmitted to disk.
  - When both tasks are finished, the roles of the buffers can be exchanged.
- The actual management is done by the OS.

Other Buffering Strategies

- Multiple Buffering: instead of two buffers any number of buffers can be used to allow processing and I/O to overlap.
- Buffer pooling:
  - There is a pool of buffers.
  - When a request for a sector is received, OS first looks to see that sector is in some buffer.
  - If not there, it brings the sector to some free buffer. If no free buffer exists, it must choose an occupied buffer. (usually LRU strategy is used)