CSCU9V4
Systems

Systems lecture 12

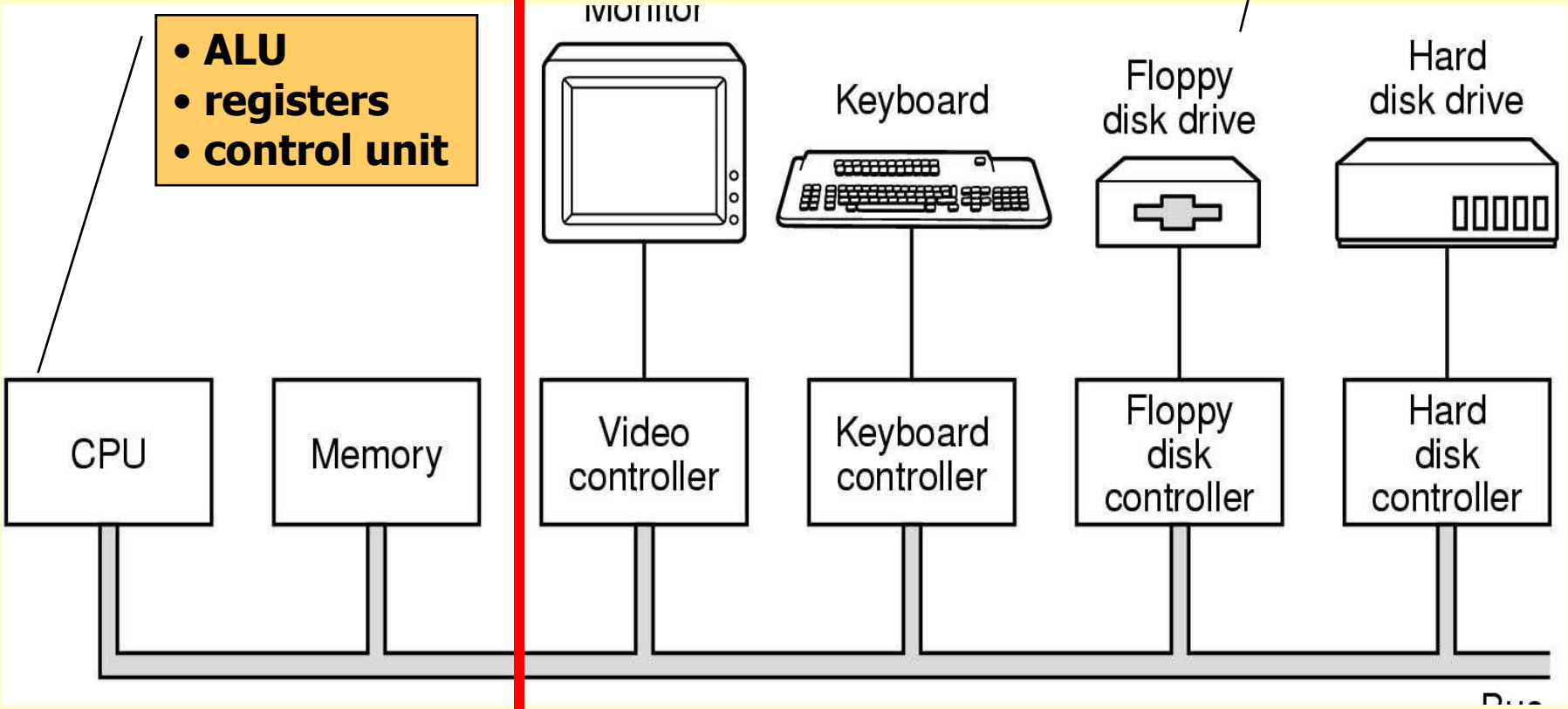
Computer Organisation

Peripherals 1

Peripheral devices and interfacing

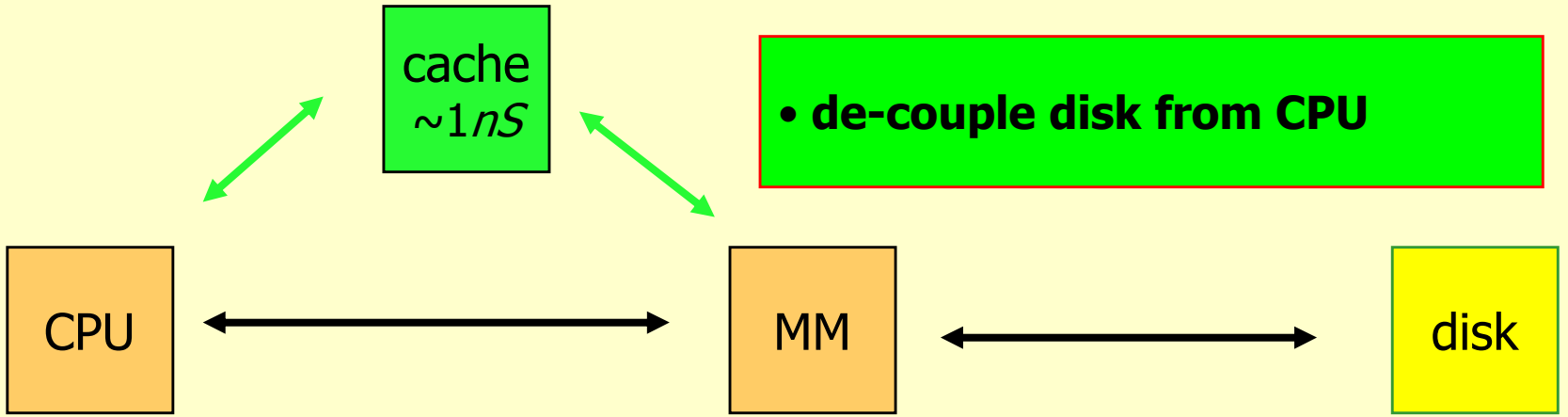
- computers need to communicate with the outside world
- consider a simple PC
- how do we *interwork* the peripherals with the CPU?

very different *requirements*



Issues: *different speeds*

- peripheral data rate << CPU/Memory data rate
- we want to allow peripheral data transfer without slowing the system down more than is necessary
- registers – less than 1nS (<1/1,000,000,000 second):
- main memory - a few nS
- disk - 10mS (about 1/100 second)



Issues: Latency and transfer rate

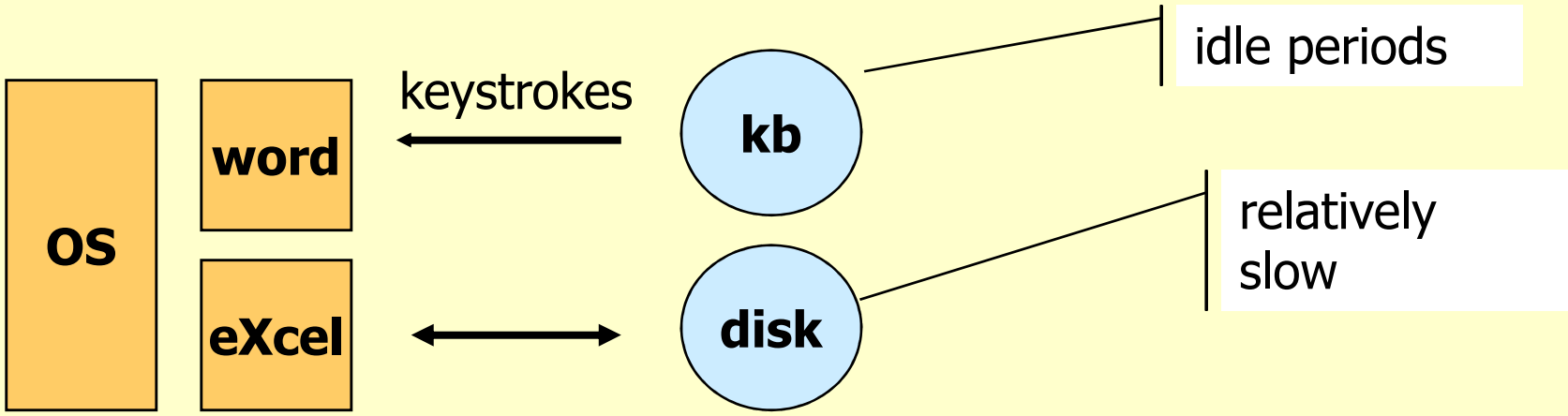
- Latency: time between request for, and start of receipt of, data
- Transfer rate: rate at which data is received, once it starts to arrive

- Want latency to be as small as possible!
- Want transfer rate to be as fast as possible

- For cache memory and registers, latency is very low, and transfer rate very high (latency usually sub nanosecond, transfer rate greater than 1word/nanosecond))
- For main memory, latency is low (a few clock cycles), and transfer rate is high (1 word/clock cycle) (latency might be 10 nanoseconds, transfer rate near 1 word per nanosecond)
- For disk, latency is high (0.01 seconds or more), and transfer rate is medium (100 to 1000Mbits/second)

Issues: *concurrency*

- in most cases we do not want the processor waiting on peripherals
- we want the peripherals to operate **concurrently**
- for example, a single-user PC running a word processor will appear to be idle when waiting for more keystroke input.
- however, the user will not expect it to stop accepting input (or displaying output) when it is accessing the disk,
- there may be ongoing background tasks.

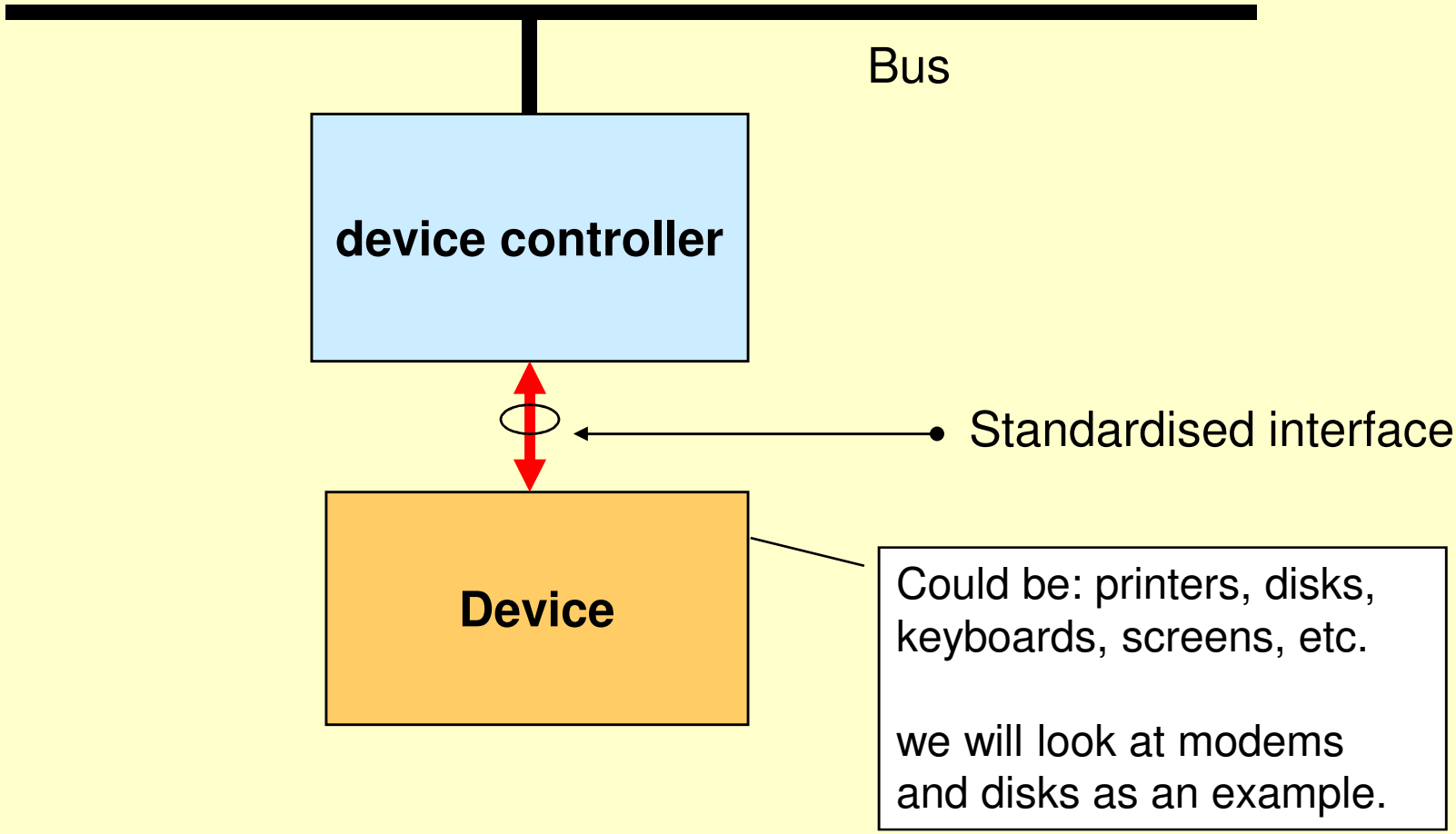


Issues: *matching electrical characteristics*

- the electrical characteristics of the information for the peripheral are usually different from those of the CPU/Memory or I/O bus
- CPU/memory buses are optimized for speed over a short distance
- the peripheral connections must compromise
  - may be further away
  - subject to more radio interference
- the differing electrical characteristics are reflected in different physical characteristics for a peripheral cable
  - Older printers usually use a D-type plug
  - USB devices use a small USB connector
  - A particular peripheral interfaces normally conforms to particular (international) standards (more later).

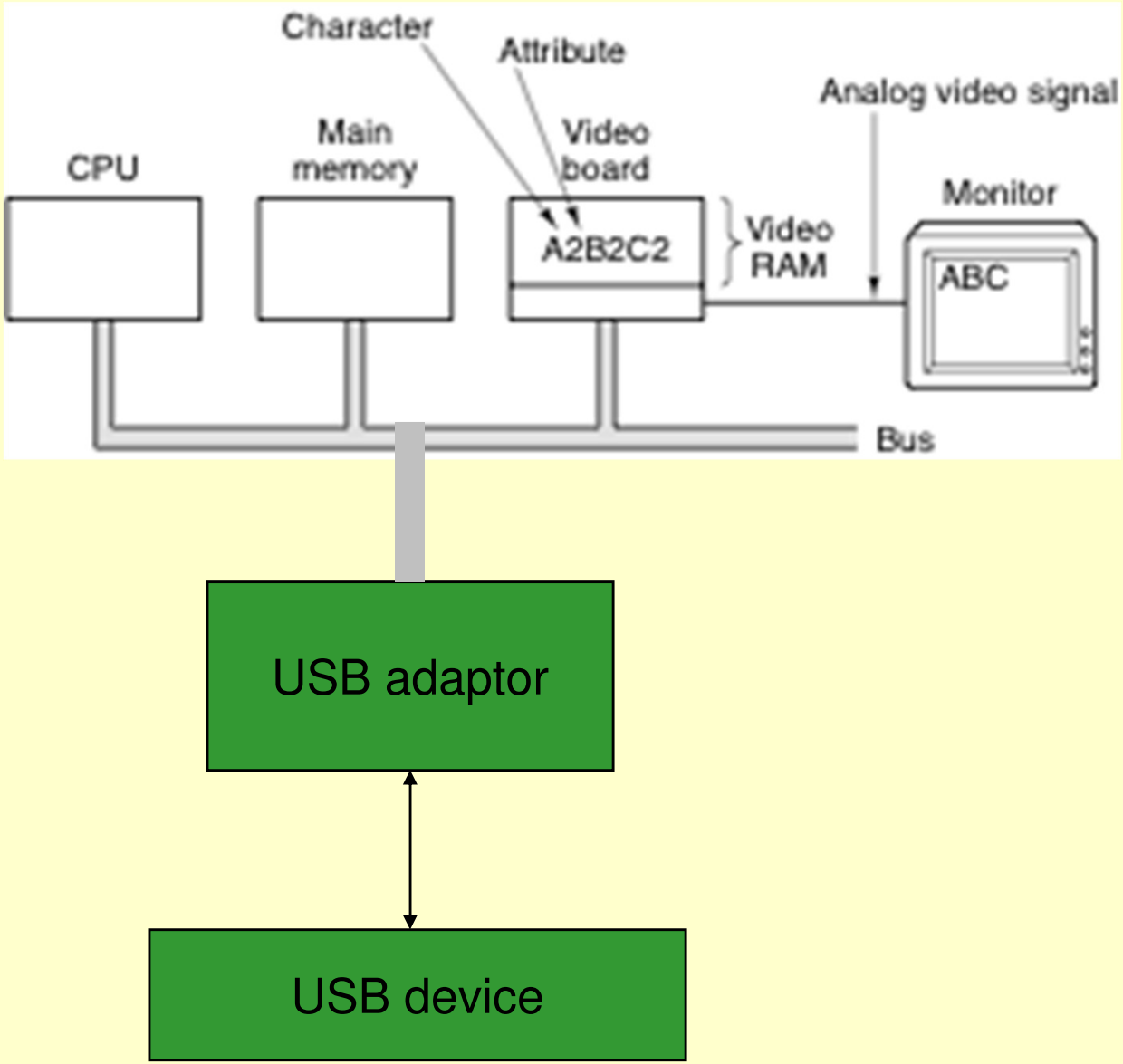
Interfacing standards (1)

- the interface provided for the peripheral is generally *standardised*.



Interfacing standards (2)

- **for example …**



Interfacing standards (3)

- by adopting internationally agreed standards, computer manufacturers allow other manufacturers' equipment to be attached to their machines.
- examples of standards are:

USB1, USB2, USB3	Mice, keyboards, Printers, etc.
FireWire	Consumer Equipment
EIDE, ATA	Disk interface, CD interfaces
RS232, RS423	Modem and terminal interfaces (old)

- for multimedia computers, interfaces to external equipment must conform to existing standards for video (usually colour video) and audio (either for digital or analogue audio input and output).
- in addition, most modern computers are provided with standard LAN interfaces.

On standards

- standards should cover all aspects of the interface to the external equipment:
  - physical connector
  - electrical signal levels
  - coding of information in each line
  - usage of lines

- if a single standard covers all of these, there is a good chance that equipment built to that standard will actually work across a wide range of machines!
  - Particularly important to equipment manufacturers

Serial interfacing

- many simple peripherals have relatively low data rates, and are often connected to a *serial interface*.
  - Actually, even some quite high data rate devices work with a serial interface
- that is, data is sent from/to the peripheral *one bit at a time*.
- in this way, the cable connecting the peripheral to the computer can be reasonably thin, as there is only one data line
  - (usually two: one in each direction) …
- + possibly some control lines.

- E.g. USB devices, FireWire, RS232, RS423, Thunderbolt

Example serial peripherals

Keyboard: Up to about 10 characters/second.
Each is 7 or 8 bits long

Mouse: Up to about 100 values per second from moving the mouse, plus button presses.

Modem: Depending on the type of modem, from 6000 to 500,000 or even 1-40 Million bits/second (broadband)

Anything attached to USB!

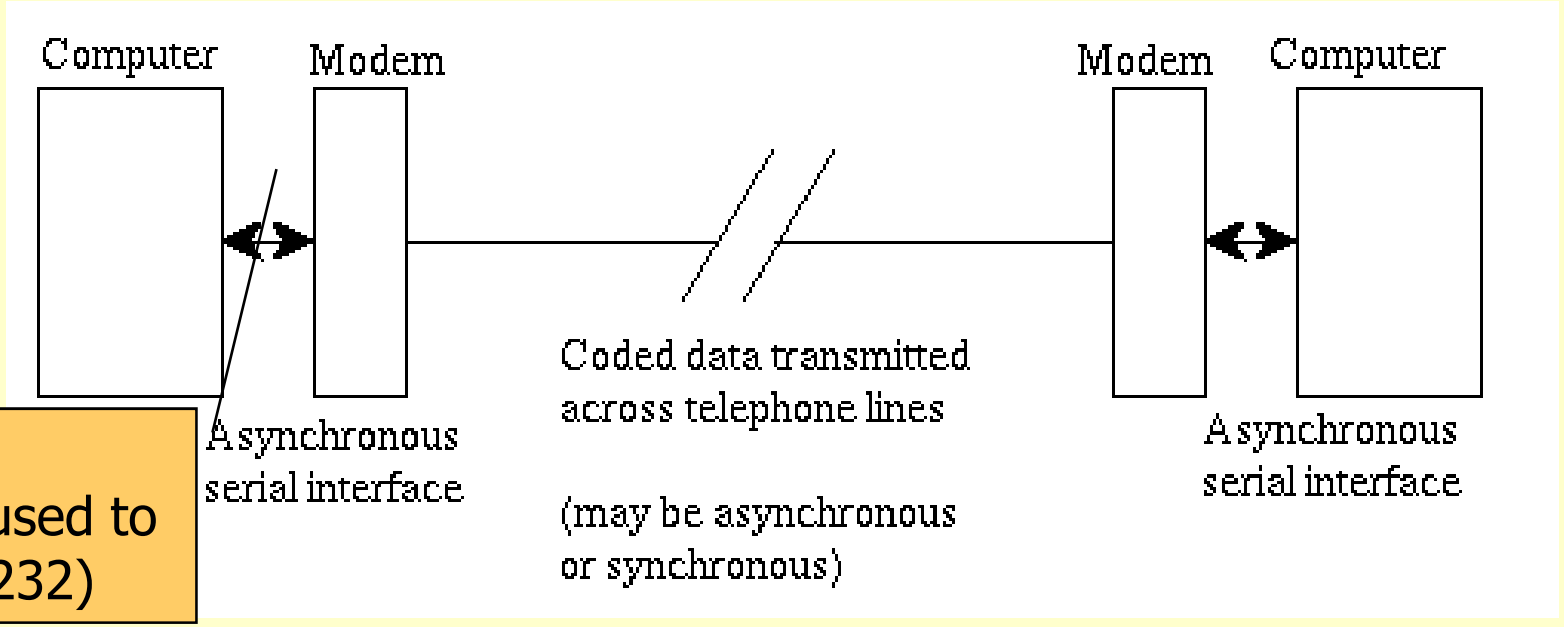
Many printers can use serial connections.

Digital still cameras have serial connections.

Also MP3 (type) players have serial connections.

Modems and serial interfaces

- Modems (MOdulator/DEModulators) are used to transfer information across telephone lines.
- they are one of the main applications of serial interfaces.



Often USB (used to be RS232)

- when a modem is used with a computer serial interface there are usually additional control lines which are used to show (e.g.) that the line is still open, that the remote end is ready to send data.

Modems (2)

- The rate at which data can be sent down a telephone line depends on the quality of the circuit.

- For a non-broadband modern digital line, this can be 56 Kbits/second.
- For broadband, 500,000 to 50,000,000 bits/second.

- Modern modems are complex devices: they often monitor the state of the line being used, and adjust the speed at which the data sent down the line is sent , and even correct errors in transmission that are introduced by noise (etc.) on the line.

- clearly, the way in which this is done must be standardised,
  - (otherwise, modems from different manufacturers cannot be used together).

Disk interfacing

- for many years, magnetic disks have offered the cheapest form of non-volatile secondary storage.

- costs for memory are normally counted as ***cost/bit***:

  RAM costs is about £20/Gbyte, or £0.02/Mbyte

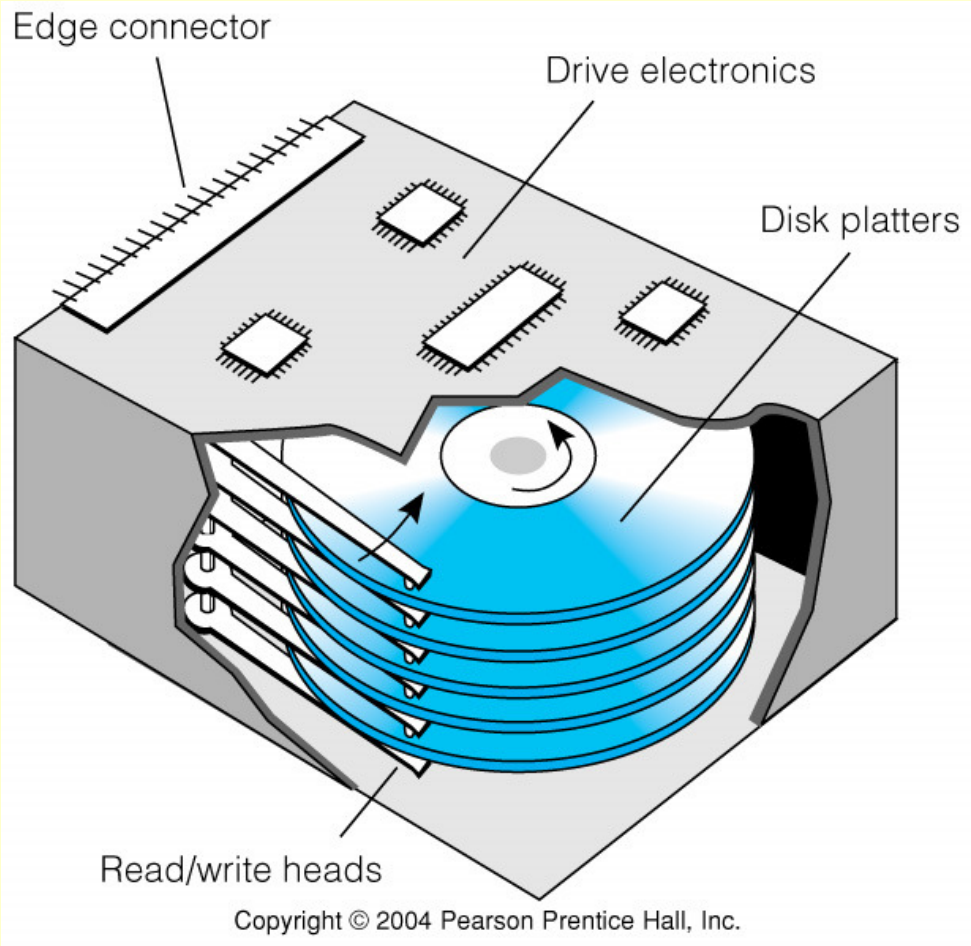
  Hard disk costs about £0.1/Gbyte, £0.0001/Mbyte.

  (priced at £40 for 2Gbyte memory, £100 for 1Tbyte disk)

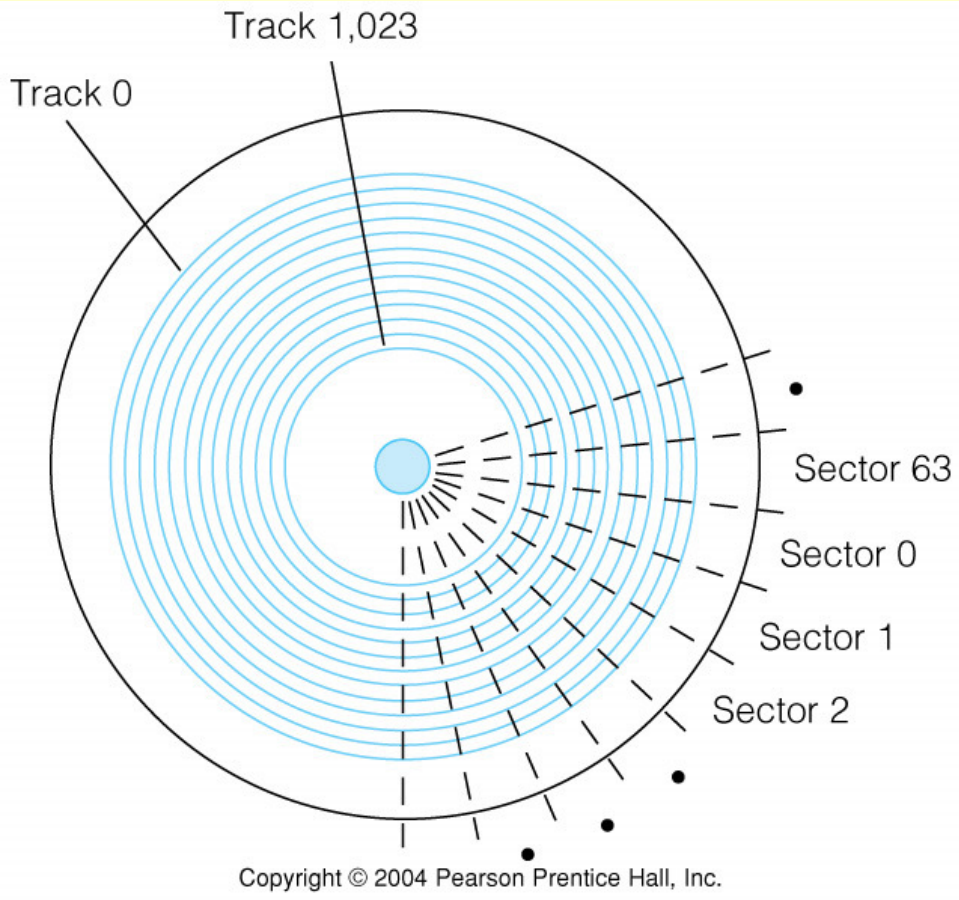
- both of these costs continue to fall (on average)…

- thus disk memory is approximately 200 times cheaper than main (semiconductor) memory at current prices.
  - **This has changed over the last few years - semiconductor memory has become a traded commodity, and disk prices keep falling.**

Cutaway View of a Multi-Platter Hard Disk Drive



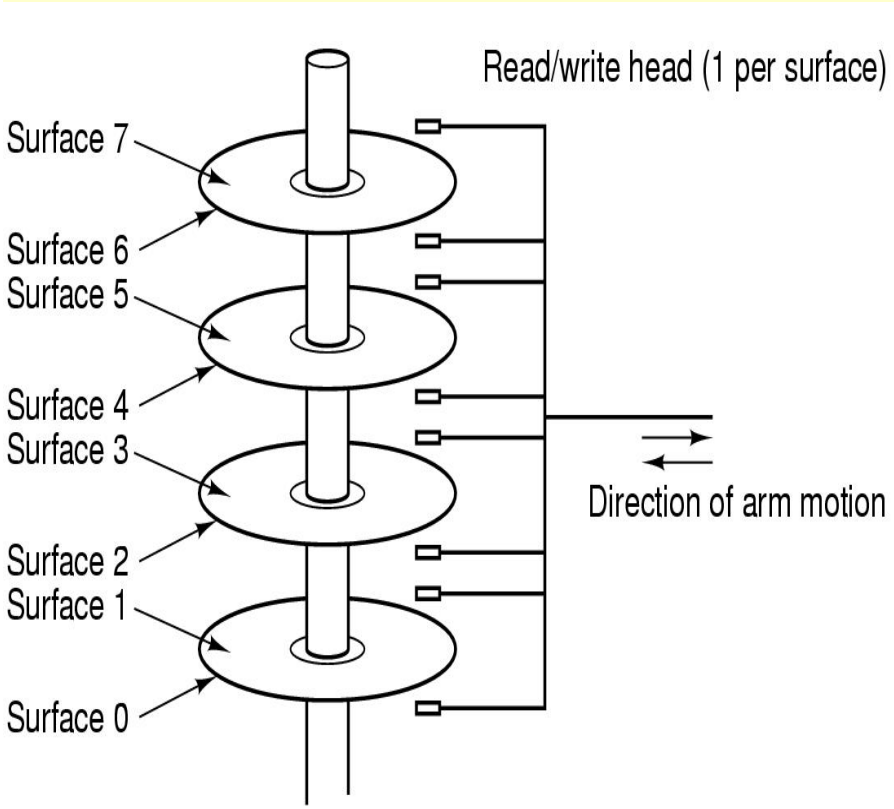
Disk Track and Sector Organization

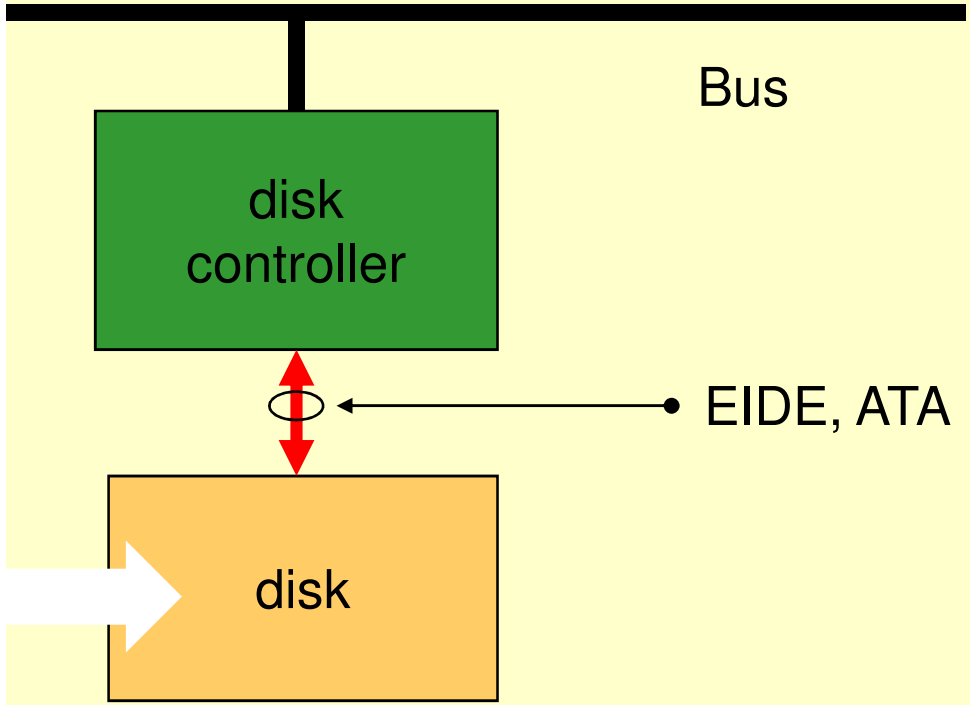


- An integral number of sectors are recorded around a track
- A sector is the unit of data transfer to or from the disk

Disk controller

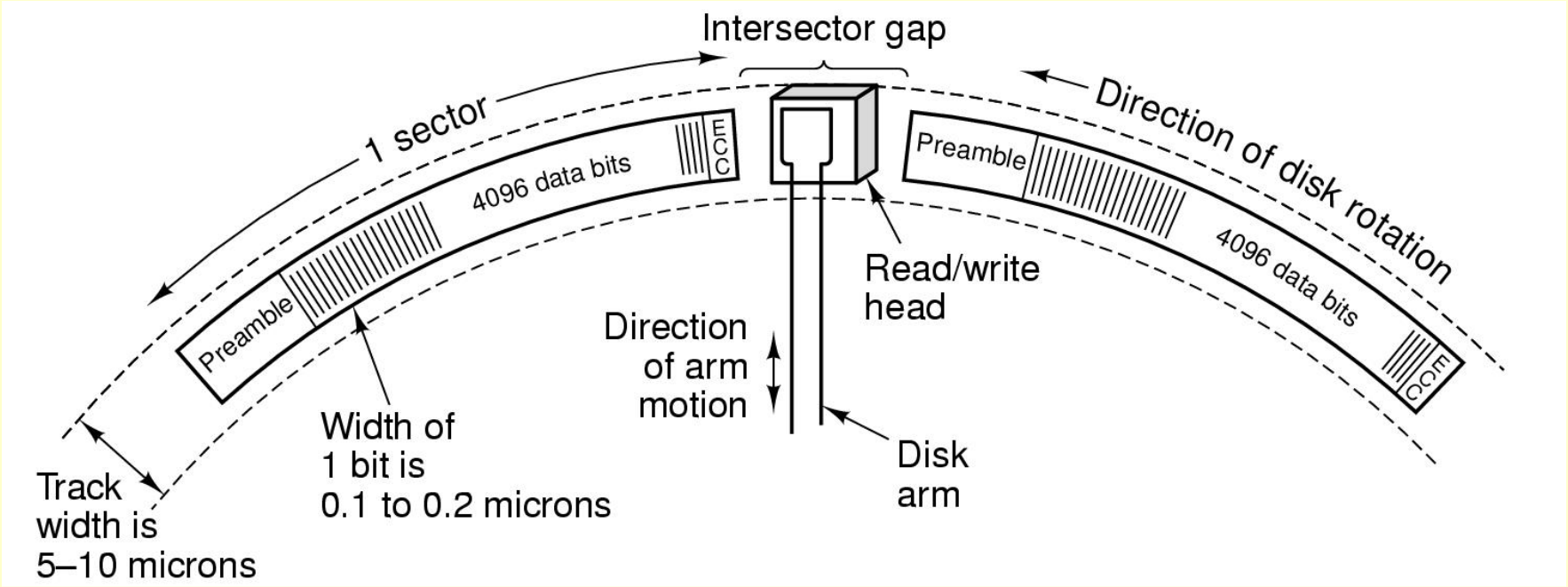
- **associated with each disk drive is a *disk controller* (a processor chip)**
  - accepts commands like ‘read’, ‘write’, ‘format’
  - controls arm motion
  - detects and corrects errors
  - converts data between streams of bytes and serial bit streams.





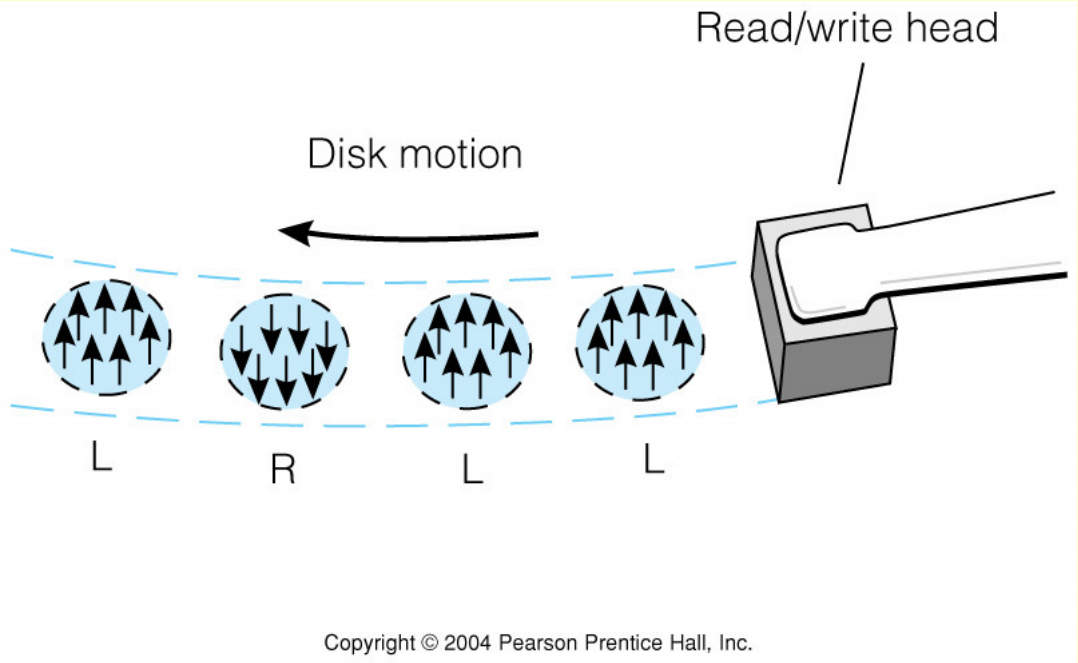
Disk organisation

- a disk unit contains one or more physical disks. On each disk there are two *surfaces*. On each surface there are a number of *tracks* (concentric circles), and each track is composed of a number of *sectors*.
- there is additionally the concept of a *cylinder*: all of the tracks of a certain size, one per surface, form a cylinder.
- data is normally stored in sectors: and a whole sector is read or written at a time.
- data transfers are through *read/write heads* (one per surface).
- disks rotate at a fixed speed, continuously (e.g. 5400, 7200 or 10000 rpm).



Individual Bits Encoded on a Disk Track

- Inside tracks are shorter & thus have higher densities or fewer words
- All sectors contain the same number of bytes
  - Inner portions of a platter may have fewer sectors per track
- Small areas of the disk are magnetized in different directions



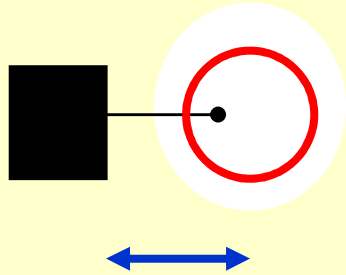
- Change in magnetization direction is what is detected on read

Accessing data on a disk (1)

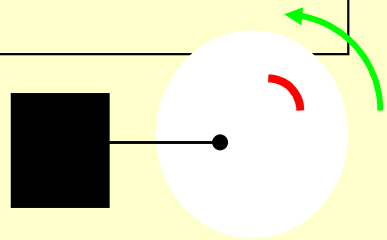
- unlike main memory, the data on a disk is not all immediately accessible. Disk I/O involves:
  - waiting for the data to be under the disk read/write head,
  - reading (or writing) the data once it is under the head.
- the wait for the data has *two parts*:

1: ***seek time***

- this is the time it takes to move the read/write head so that it is positioned on the correct track.
  - this will depend on how far the read/write head is from the correct track in the first place.
- usually this seek time is between 3 ms and 50 ms.



Accessing data on a disk (2)



2: ***rotational delay*** (or ***latency***).

- this is the time it takes for the required sector to appear under the read/write head.
- if the disk rotates at 7200 rpm, one rotation takes 1/120 second = 8.3 ms.
- on average, latency will be one half of a revolution, that is 4.2ms.

- ***actual waiting time = seek time + latency***

- so speeding up access on a disk can be achieved by
  - making it rotate faster, or
  - moving the read/write head from track to track more rapidly.

Data transfer

- once the head is in position over the start of the sector to be read/written, data transfer is fast.
- the sector is read/written as it passes under the head.
- an entire track can be read/written in the time it takes for the disk to rotate once.
  - The transfer rate between the disk and the read/write head is fixed by the rotational speed of the disk, the number of bytes/sector, and the number of sectors/track.
- a single data transfer operation may transfer as little as a single sector, or several entire tracks (on a cylinder).
- (an entire cylinder can be accessed without moving the read/write heads.)

Data transfer (example)

- say:
  - there are 100 sectors/track,
  - each sector contains 1024 bytes, and
  - the disk rotates at 7200 rpm.
- then one sector is under the head for

  (60/7200) * 1/100 seconds = 8.3/100 ms = 83 microseconds.
- this is the time for the transfer of one sector. So the transfer *rate* is

  1024 * 1000000/83bytes/second , i.e. about 3 Mbytes/second.
  - Note that this is the transfer rate from the disk surface to the head electronics. Normally, the disk controller has local storage, so that the data is buffered at the disk controller, and then transferred to the main memory.
- actual disk controllers and disks have data transfer rates from about 3 Mbytes/second up to 20-30 Mbytes/second.

Software requirements (1)

- a program will manipulate *files* ….
  - (and data or records from files)
- … not tracks and sectors.

- one of the functions of the operating system is to perform a mapping from the **logical** view of the disk
  - as directories, files, etc,

- to the **physical** view of the disk
  - as cylinders, tracks, and sectors.

Software requirements (2)

- programs need to
  - Create a file
  - Open a file
  - Close a file
  - Delete a file
  - Read a record (or a byte)
  - Write a record (or a byte)

- we need to consider how these requests become transformed into requests to read and write appropriate sectors of a disk.

Software requirements (3)

- an *unformatted* disk contains a number of sectors: these are all identical.
- the formatting operation sets up a disk catalog, and puts some data on to the disk so that the OS can put files and directories on it.
- different operating systems use different disk formats.
  - So you can't just take a MAC OSX disk and read it on a PC running Windows or Linux (without adjusting how it is read).
- the general concept of using a catalog of some sort to find the beginning of a file, or to allocate disk sectors to a new or expanding file is common across different operating systems.
- the basic idea is that the name of the file may be looked up in the disk catalog, to find a sector number.

End of lecture