

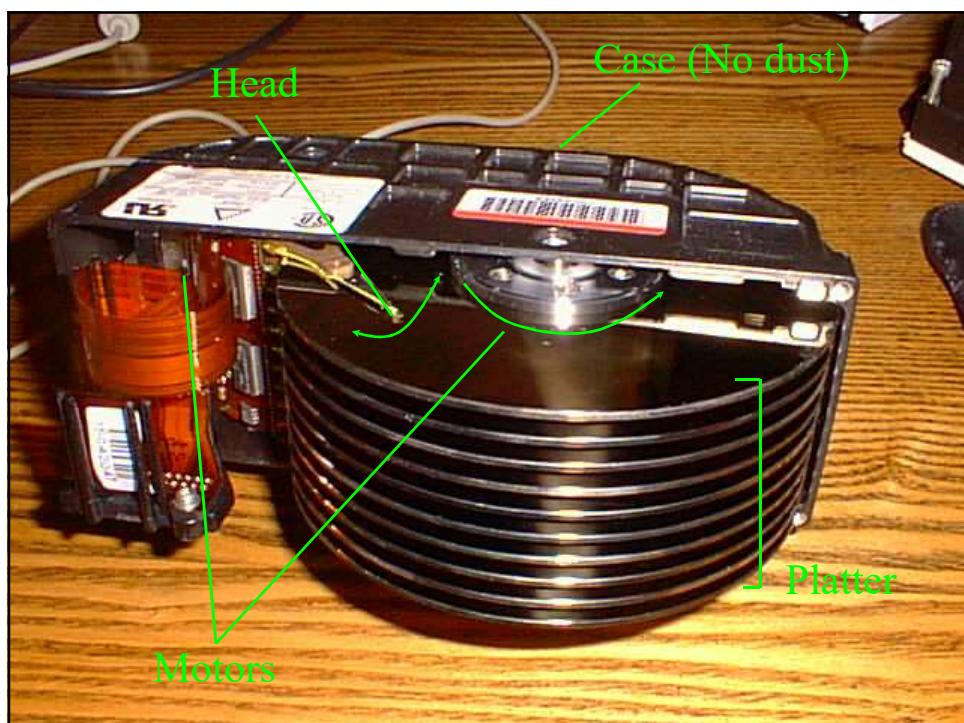
CS253Architectures II

Lecture 13

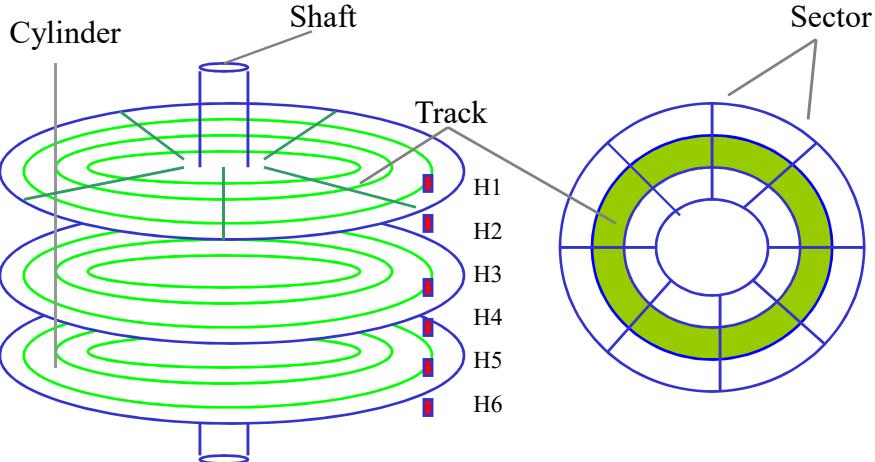
The hard disk

CD-ROM

Charles Markham



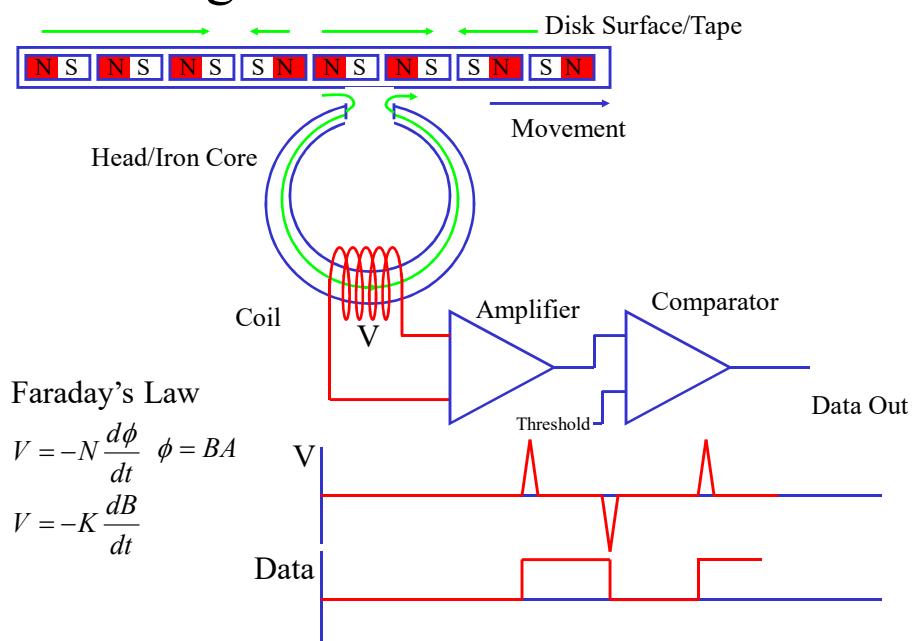
Disk Layout



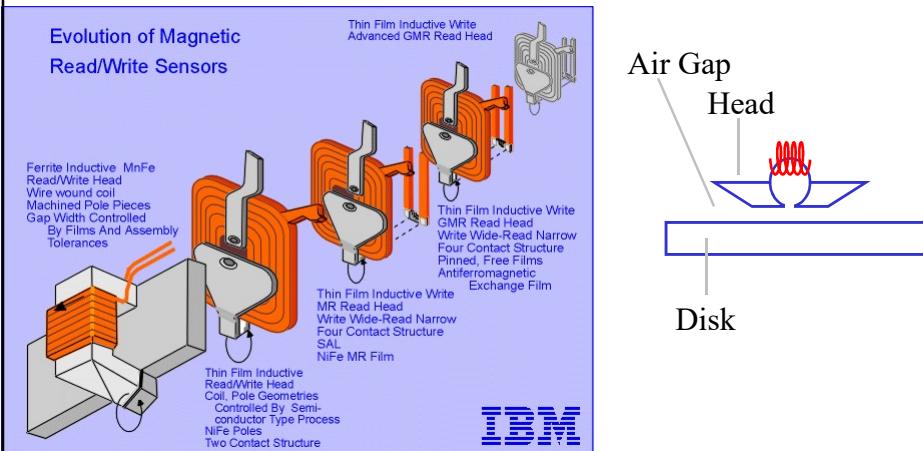
Read data from a sector, given the cylinder, head and track number.

512 Byte block x Cylinders 12495 x 16 Heads x 63 sectors = 6GByte

Magnetic Read/Write Head

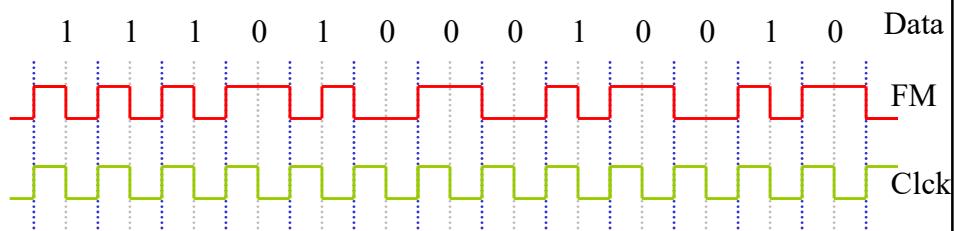


Heads



On floppy disks the heads touch the surface of the disk, on hard disks they fly over the surface. The heads are so close that even smoke particles could cause them to 'crash' into the disk surface. Hard disks are sealed into their case.

FM Data Coding



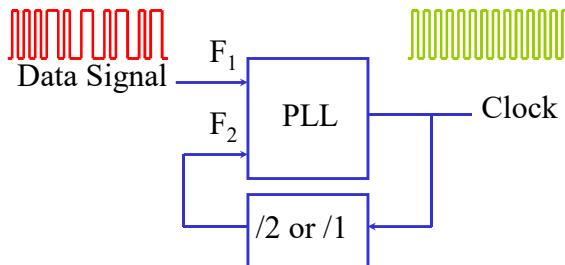
FM (Single density, F2F): Data bit 1 makes signal change level, Data bit 0 signal does not change, signal changes level after each time slot. Note that there are 1 or 2 edges per time slot.

How do you create the clock if you only have the signal?

Phase locked loops

As the disk rotates data is read from the disk. A clock (square wave oscillator) is synchronised with the rotation of the disk. Using the clock and data signal it is possible to know from where you are reading the data. The device used to synchronise the clock is known as a phase locked loop or PLL.

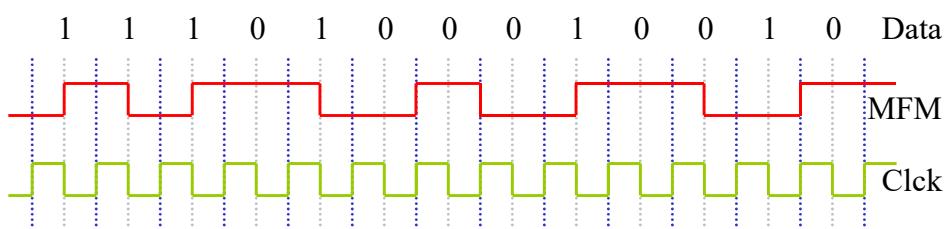
A PLL adjusts its output frequency so that difference between the difference between the input frequencies is zero.



Note PLL can freewheel past missing pulses and stay locked to the data signal.

MFM Data Coding

How do we reduce the number of transitions and store the same amount of data?



MFM (Double density, Modified Frequency Modulation):

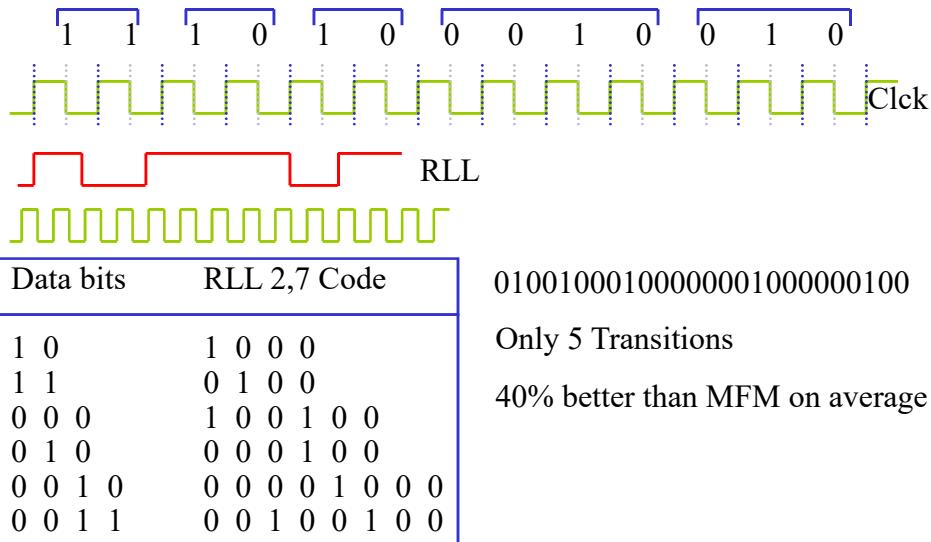
Create a transition for all the data bits of value 1,

Transition at start if current data bit is 0 and previous data bit is 0.

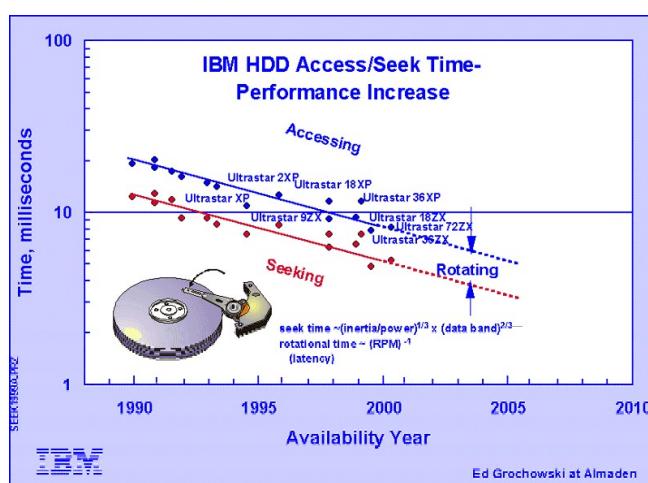
Roughly half the number of transitions, sequence packed closer together on disk, double density.

RLL Coding

RLL, Run Length Limited



IBM announced a GMR head which can read data at 35.3 Gbits/in²



Access time 6ms, 0.006 seconds to read data from any point on the disk. 166RPS or 10,000RPM!. Disks run on air bearings with brushless motors (stepper).

Some performance parameters.

Access Time: Time between a request for data and receiving the data, typically 20mS.

Seek Time: Time taken for heads to move to a specific location, typically 12mS.

Latency Time: Time taken for PLL etc to start reading data from the track, 8mS.

Transfer Rate: Rate at which data can be read from the disk at a sustained rate. 10Mbits/Sec (HD), 1Mbit/sec (Floppy)

Track/Sector formats

So far we have only considered how the data is physically stored on the disk. The sectors contain ID, Location information, Saved Data, and checksum information.

The checksum is used to see if data was successfully written to the disk. If an error occurs the sector is marked as ‘bad’ and the data is written to a different sector.

Track 1

Preamble 46 bytes	Index Add. 1 bytes	Gap 36 bytes	Sector 1	Sector 2		Sector 26	Postamble
----------------------	-----------------------	-----------------	-------------	-------------	--	--------------	-----------

Sector 2

Id Add. 1 bytes	Sector I	Gap	Data A. 1		Data	33
--------------------	----------	-----	--------------	--	------	----

Some performance parameters.

Access Time: Time between a request for data and receiving the data, typically 20mS.

Seek Time: Time taken for heads to move to a specific location, typically 12mS.

Latency Time: Time taken for PLL etc to start reading data from the track, 8mS.

Transfer Rate: Rate at which data can be read from the disk at a sustained rate. 10Mbits/Sec (HD), 1Mbit/sec (Floppy)

Track/Sector formats

So far we have only considered how the data is physically stored on the disk. The sectors contain ID, Location information, Saved Data, and checksum information.

The checksum is used to see if data was successfully written to the disk. If an error occurs the sector is marked as ‘bad’ and the data is written to a different sector.

Track 1

Preamble 46 bytes	Index Add. 1 bytes	Gap 36 bytes	Sector 1	Sector 2		Sector 26	Postamble
----------------------	-----------------------	-----------------	-------------	-------------	--	--------------	-----------

Sector 2

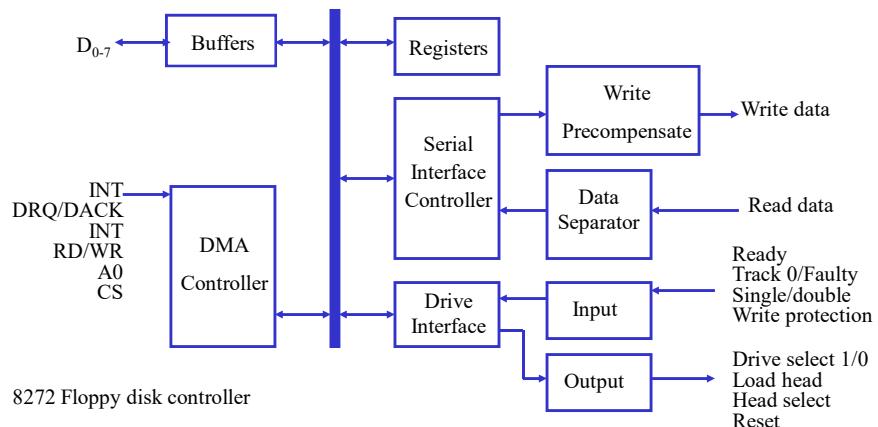
Id Add. 1 bytes	Sector I	Gap	Data A. 1		Data	33
--------------------	----------	-----	--------------	--	------	----

This is put on the hard-disk when you format it.

Disc drive controller

Synchronising the mechanical movement of the head and platters, decoding the data and providing an interface to the PC bus is done through custom IC known as disk controllers.

The disk controller shown below is typical of that used for the floppy disk drive.



Error detecting and correcting

Remember the phone number is 086-5432123

However lets say you write it down 086-54321?3 and one of the numbers is obscured.

Can we identify the missing number?

We could just repeat the number twice when we write it down.

But can we do better?

Try adding up all the digits and send the total also 086-54321?3 + Checksum=32

So we can add the digits we have $0+8+6+5+4+3+2+1+?+3 = 30$ then take away from the checksum $32 - 30 = 2$ gives the missing number.

We could use this approach to detect if one of the digits was wrong rather than missing but can't correct it....

Multidimensional parity-check code (MDPC) was used to encode 16 data bits into 25 transmission bits.

Received bits shown below.

Data	0	1	0	0	0	J
Parity	1	0	1	1	0	I
	0	1	0	0	1	H
	0	1	1	1	0	G
	0	1	1	1	0	F
	A	B	C	D	E	

Note: The bit at BH is in error as the parity on the horizontal and vertical columns is different to that expected.

Disk Formatting

The Floppy Disk (or any boot disk) contains data starting at Track 0, Sector 0 organised as follows.

Boot record - variable size,

File allocation table - known as the FAT

Copy of the file allocation table

Root directory

Data area

Boot-DOS(disk operating system)-lifting the compute up by its bootlaces.

FAT/Clusters

There are very many sectors on a disk

The File Allocation Table stores the location of (Track, Sector) of the start of each cluster stored on the disk.

Win 3.1, Win95 950/950a uses FAT 16

Win 95 950/950c and Win 98 and NT uses FAT 32 or FAT 16

Each cluster contains up to 32,768 bytes

Sectors are grouped to form clusters ($32,768/512=64$ sectors/cluster).

FAT 16 allows $2^{16}=65,535$ clusters to be contained on one drive.

Total storage $65,535*32,768 = 2,147,488,648$ Bytes

Sets a 2 Gbyte limit on drive size.

Q: How do you use a bigger disk?

Disk Partitioning

A hard disk that exceeds 2Gbyte can split into two halves and treated as two separate disks by the OS (even though there is only one physical disk drive). Thus a 6Gbyte disk drive using FAT 16 would require three logical drives (C:, D:, E:) to access the disk.

Q: how much disk space is required to store a 1K file on 2Gbyte disk using FAT 16?

A: At least 1 cluster is required to store the file (since the disk is 2Gbyte each cluster must contain 32768 bytes. Thus 31744 bytes are wasted and 1024 stored (3% used)).

FAT 32 Solves this problem to some extent.

FAT 32

FAT 32 can reference $2^{32} = 4,294,967,296$ clusters

Each cluster need only be one sector in size for the 2Gbyte disk

Q: What is the cluster size of a 10Gbyte disk using FAT 32?

A: Technically

10Gbyte should equal $10 \times 1024 \times 1024 \times 1024 = 10,737,418,240$

in practice manufacturers mean 10,000,000,000.

Again $10,737,418,240 / 4,294,967,296 = 2\dots$ bytes

one cluster per sector.

The Root Directory

The root directory contains 32bytes of data for each file stored on the disk .

Byte Number

0-7 Filename

8-10 Extension

11 File Attribute, 1 Read, 2 Hidden, 4 System file,

12-21 Reserved

22-23 Time file was created

24-25 Date the file was created

26-27 Starting cluster number

28-31 File size in bytes.

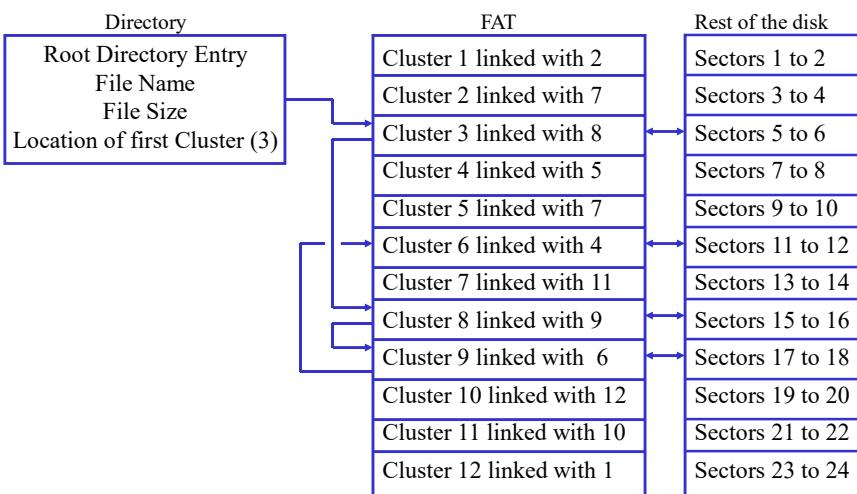
Clusters to Files

The root directory identifies the first cluster in the FAT for a specific file.

The first cluster entry in the FAT will contain the cluster number for the second cluster. The process continues until enough clusters are linked to contain the file size.

The root directory and FAT form a linked list that refers to a set of clusters that contain the file data

Summary



EIDE/SCSI

IDE (Integrated Disk Electronics)

Enhanced IDE (EIDE also called ATA-2). Classic IDE supported two hard disks of 528 megabyte or less. EIDE allows four devices, including a mixture of disks, tapes, and CD-ROM, and the hard disks can be larger.

SCSI (Small computer systems interface, scuzzy), 50 wire cable

SCSI is used by Macintosh computers, RISC workstations, minicomputers, and even some mainframes and is available on PC's.

Allows many devices to be chained together.

Microprocessor does not need to operate if hard disk is being backed up on a tape streamer.

SCSI Cables need terminating!

RAID

RAID (Random Array of Inexpensive Disks)

Allows faster disk access, improves reliability and removes down time for service.

Data is striped across a number of disks. Extra parity information is stored on the disk.

Since data is read/written to the disk in parallel data rates can be higher.

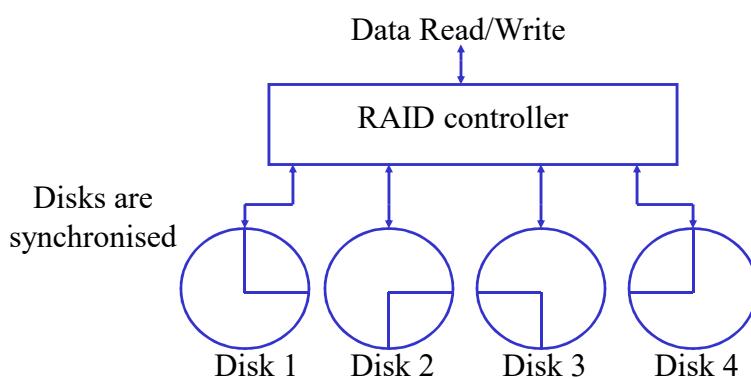
Disk can be removed and replaced without loss of data. The missing disk is rebuilt from the parity information stored on the other disks. This is known as a hot-swap.

RAID

The following RAID levels are available

- Level 0 Striping without parity across four disks, gives increased speed but no improvement in data I integrity.
- Level 1 Disk mirroring, no striping, maintains a copy of one disk on another disk.
- Level 3 Striping, with an extra dedicated parity disk. Only 20% of the disk is used to store redundant information.
- Level 5 Data and parity are striped across 5 disks.
- Level 10 Striped data across two disks, disks are mirrored.

Raid controller



File Delete on FAT 32/FAT16

When you delete a file from a disk the first character of the name of the file is changed to a ‘?’ character. The clusters used by the file can then be used by other files for storage. It should be noted that most deleted files stay intact for quite a while and are easily undeleted.

To undelete files type LOCK C:<ret>, a:undelete, UNLOCK C:<ret>. Undelete.exe is readily available as part of DOS 5.0.

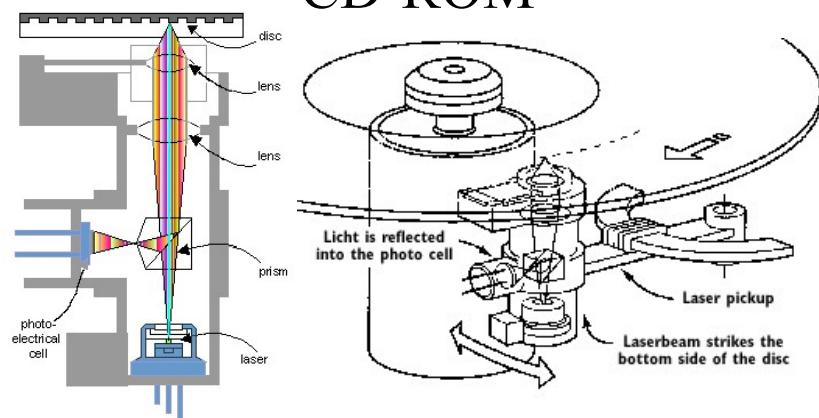
An empty Trash-can can be restored using this approach.

cd c:\recycled, ... (Note recycled is a hidden directory in dos)

Deleted files can pose a threat to security since deleted files may not have the same protection.

Use erase to permanently remove disk data.

CD-ROM



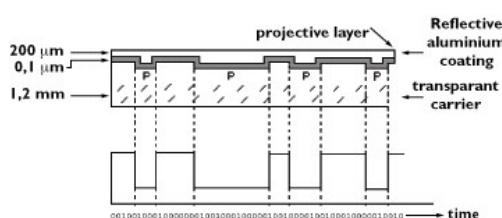
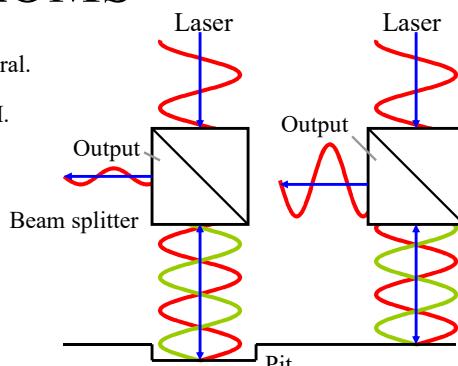
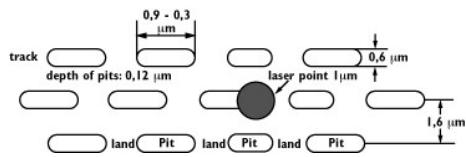
DVD Digital Versatile (formerly video) Disc single layer 4.7 GB of capacity, CDs 680 MB capacity or 17.0 GB capacity of the double sided, dual layer DVD.

CDs can be mass produced by mastering/pressing or created individually using CD-R, CD-RW technologies.

CD-ROMS

20,000 tracks on a CD organised as a single spiral.

Constant linear velocity scanning 200-500RPM.



<http://www.os.philips.com/cd/cd-rom>

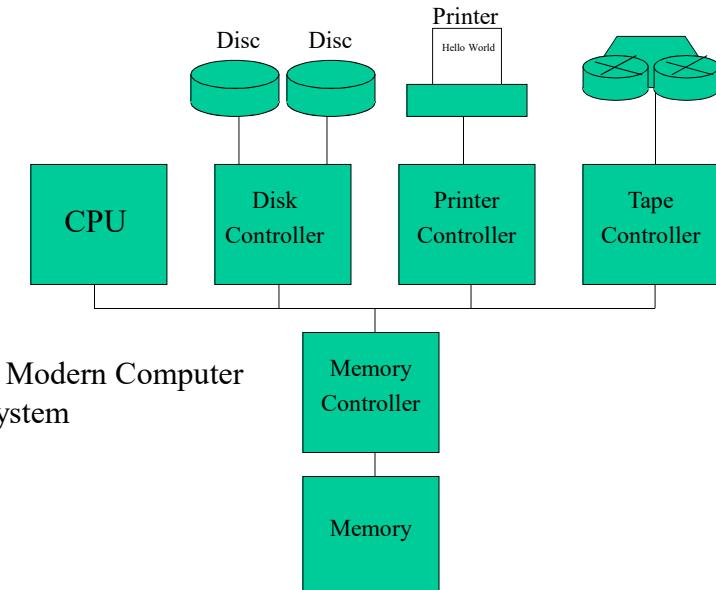
CS253 Architectures II

Lecture 13b

Hardware protection for run time applications

Charles Markham

Computer System Operation



Turn on a computer

Bootstrap: Initialises registers.

Loads Kernel into memory (core of the OS).

OS waits for events to occur, notified via an interrupt.

Software or Hardware interrupt occur.

Each interrupt has a service routine.



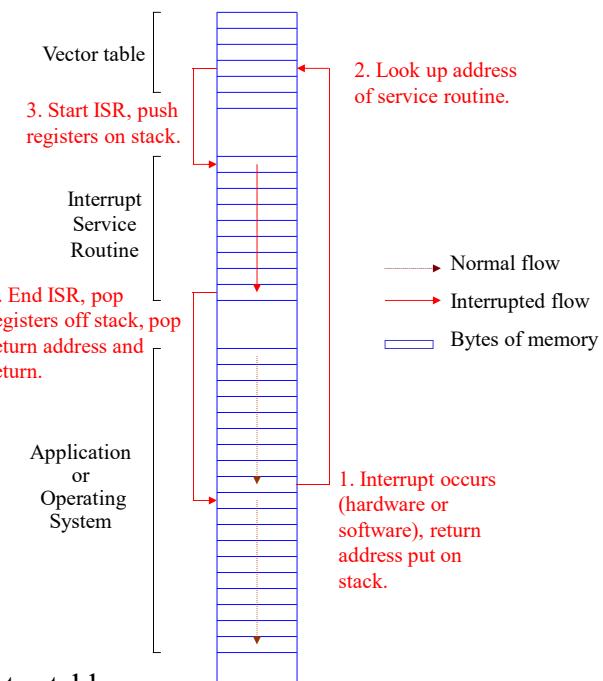
Interrupts

Interrupts are disabled during the processing of an existing interrupt.

The 8259 allows an interrupt to be interrupted based on priority.

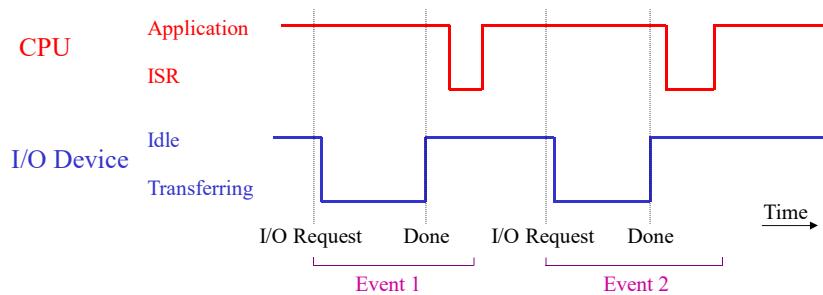
The aim is to avoid having lost interrupts.

UNIX and DOS use a vector table.



Interrupts

When an interrupt occurs control is handed back to a service routine contained in the operating system. If the service routine was called through a vector table it will service the hardware immediately. Sometimes more than one piece of hardware can cause an interrupt so the service routine has to poll devices using I/O to see which needs servicing.



I/O request causes hardware to start obtaining data, the rising edge of the I/O causes interrupt to occur indicating that the hardware needs servicing. Service routines starts shortly afterwards to obtain data acquired by the hardware. Normal operation until the next interrupt.

Interrupts, Traps and Exceptions

On an 8086 interrupts come from one of three sources.

Hardware: A signal to NMI (Non maskable interrupt line).
A signal to the INTR line (via one of two 8259PICs).

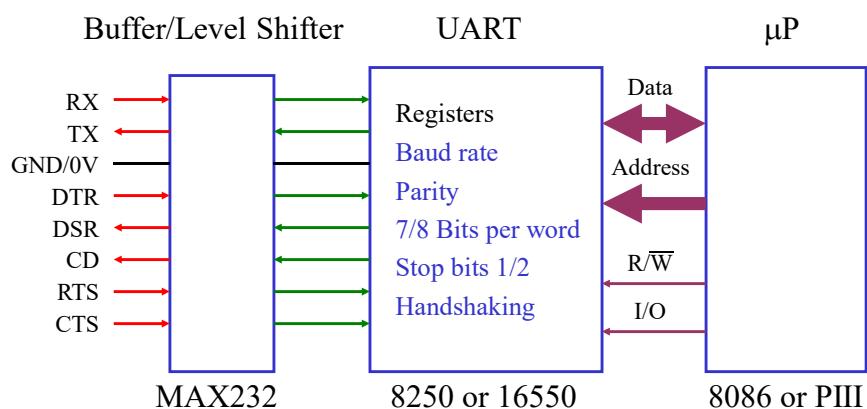
Software: A software interrupt such as *int 021h* used to print.

Errors/faults: Exception interrupts, divide by zero, parity error, single step etc

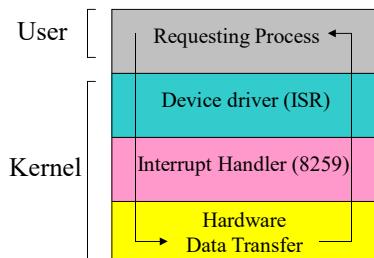
8086 checks to see if an interrupt has occurred at the end of each instruction

I/O Structure

Most device controllers contain a set of special purpose registers a buffer and some I/O. Consider the 8253 UART we studied in Architectures, the chip was configured for Baud Rate, Parity, Number of bits etc through registers, a buffer stored the incoming and outgoing characters and I/O allowed the OS to send and receive data from the device.. .



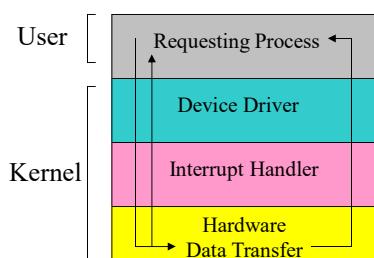
I/O Methods Synchronous



After I/O starts the CPU controls the hardware, on completion of the I/O task control is returned to the requesting process.

Note: If the CPU is only wasting time during the I/O cycle then this is an inefficient process.

I/O Methods Asynchronous



After I/O starts control is handed straight back to the requesting process. An interrupt then signals the end of data transfer from the hardware.

Note: If the requesting process can't proceed until the completion interrupt has occurred then an instruction **wait** is available that prevents the IP from increasing (similar to **J1: jmp J1**). The interrupt return control to the OS. This is inefficient because the CPU can only process one hardware device at a time.

I/O Methods Asynchronous

Best of all:

Start the I/O and continue processing user programme or OS code.

A system_call to the OS starts the I/O device and/or waits for the data.

A device status-table is used by the OS to check the status of the hardware. Each device may have a queue of requests.

If the Hardware is busy and the OS is not the OS just waits otherwise it processes other interrupts.

I/O Device interrupts the OS, OS checks device status_table, modifies the table and allows the data to be returned by the original system call.

Some hardware contains buffers that work independent of the OS.

Hardware Protection

A properly designed OS must ensure that an incorrect program can not interfere with another program. This could occur if two programs were making use of the same piece of hardware and one program fails denying the other of the hardware resource.

The OS must prevent one program from interfering with another.

If the hardware detects an error it will send an interrupt (it traps the error) and the OS will report the error, terminate the program and store a record of the memory associated with the program when it failed (dump).

Dual Mode Operation (1)

The CPU allows two modes of operation. A single bit in a registers sets which mode the CPU is running in.

User Mode: Code run on behalf of the User.

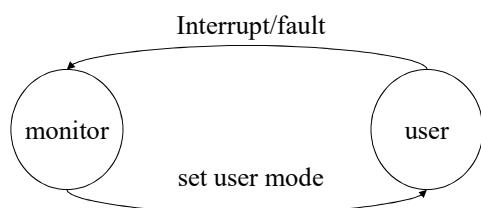
Monitor Mode: Code run on behalf of the Operating System.

In user mode you can not change to monitor mode.

The OS switches to monitor mode before handing control to user programme.

Privileged Instructions that are capable of harming other programs are available in monitor mode only.

Dual Mode Operation

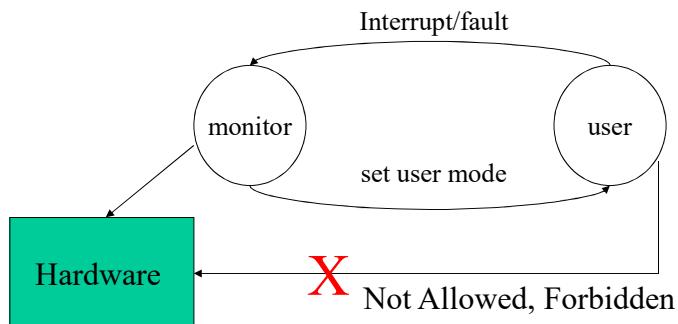


Monitor mode allows unrestricted access to the system (also known as supervisor mode or system mode).

User mode is monitored by the OS, programs must go through the operating system to make requests that may affect other programmes.

I/O Protection

All I/O instructions are privileged instructions.



It is only through the OS that the USER can interact with the hardware!

Memory Protection (2)

Must provide memory protection for Interrupt Vector Table.

Why?

Service routines are part of the OS and run in monitor mode. By re-vectoring entries in the table (as we did!!) you can jump back to a user program in monitor mode.

The Solution is Memory protection:

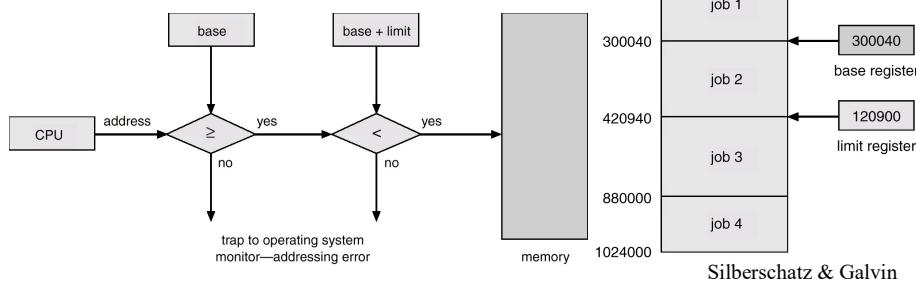
The *base register* is loaded with the lowest address in memory that is safe.

The *limit register* is loaded with the highest address in memory that is safe.

Memory outside this range is protected.

As the OS switches between each process it changes the registers to limit the processes address space.

Memory Protection



Hardware used to check memory limit bounds.

Regularly causes the ‘blue screen’ error message “General Protection Fault” in Windows.

CPU Protection (3)

If a program goes badly wrong how does the OS get a chance to regain control?

The timer interrupt fires at regular intervals (18.2 times a second on the PC).

The timer can be used to cause a *context-switch* between processes. This stops one process and switches to another, multitasking. Each process is allowed a period of time called a *time-slice*.

The timer can be used to see if the user program is running for too long.

Most systems have a separate ‘real clock’ to monitor time and date.

Finally

- ◆ Given the I/O instructions are privileged, how does the user program perform I/O?
- ◆ System call – the method used by a process to request action by the operating system.
 - Usually takes the form of a trap to a specific location in the interrupt vector.
 - Control passes through the interrupt vector to a service routine in the OS, and the mode bit is set to monitor mode.
 - The monitor verifies that the parameters are correct and legal, executes the request, and returns control to the instruction following the system call.

