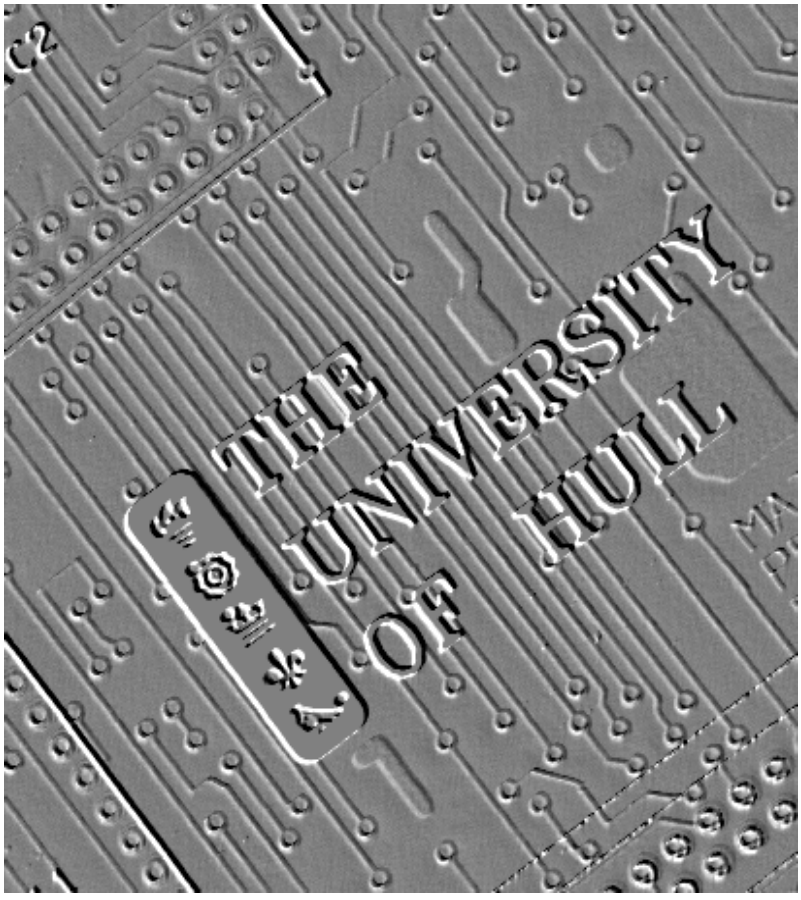
13523

Workstation Technology

Rob Miles



Department of Electronic Engineering

Contents

The author can be contacted at:

The Department of Electronic Engineering
The University of Hull
HULL
HU6 7RX

r.s.miles@e-eng.hull.ac.uk

Workstation Technology
08/10/2007 22:42:00

Data

Elements of Data

We think of computers as information processors but really they are number processors, since the actual processing unit can only work in terms of manipulating binary quantities. The way that we get useful work out of a computer is to ascribe some meaning to the numeric values that are being dealt with and then use a layer of software to "hide" the numeric nature of the work actually performed. Throughout this course we will be looking at how the capabilities of the computer hardware are made available to users via appropriate software.

Text Character Codes

An instance of where we give meaning to numbers in order to perform meaningful work is the assignment of character codes. As an example we may be running a word processing program and ask it to find us all the words which begin 'THE'. What the computer will actually do is search through its memory for locations containing 32,84,72,69 consecutively.

ASCII

The reason that these particular numbers will be searched for is that there is a standard (called ASCII - American Standard Code for Information Interchange) which gives a numeric code for each of the letters - in the case above 32 means space, 84 means "T" and so on. ASCII is only one of many codes which have been set up to represent textual information in numeric form, another is called EBCDIC.

ASCII uses numbers between 1 and 127 to represent particular characters. The code for the characters are "sensibly" organised, in that the code for "A" is less than the code for "B" and so on through the alphabet and the code for "0" is less than the code for "9". This makes the sorting of alphabetic information very easy.

The first 31 character code values are reserved for non-printing functions. These allow a computer system to send commands such as "take a new line" or "move the print head to the start of the line" to a peripheral device. Some of the non-printing codes are also used so that a device can tell another when it is unable to receive any more data, as part of a process called handshaking - of which more later.

In computer terms the values of the ASCII code, 1-127, can be held in seven binary digits (bits). ASCII characters are therefore sometimes described as "seven bit data".

Most computers store data in locations 8 bits wide, each location being called a byte. Bytes can store values between 0 and 255. This form of data is called, not surprisingly, "eight bit data". It is interesting to note that this can mean that about 12% of computer memory is never actually used!

Some peripheral makers, for example EPSON who make printers, have used the fact that extra character codes are available from 8 bit data to allow additional character sets to be supported, however these enhancements are very non-standard and only of limited use.

In short then, character codes such as ASCII and EBCDIC provide a means of computers manipulating and transmitting text. When we talk about text information being stored on a computer system we are therefore talking about a sequence of character codes, usually held in byte locations, i.e. any file of text on a computer is in fact a sequence of numbers.

Binary Data

Binary data is also held and manipulated by a computer system but it is important to make a distinction between it and text data.

The primary difference between text and binary data is that binary data, when held in the form of bytes, will contain all possible values between 0 and 255 whereas text can be expected to make use of only the first 127 values.

Because it is usually used only by the particular machine binary data is always specific to that machine, for example the binary for a compiled program for one make of computer will not work on a different one.

Binary data can also be a problem to transfer between computer systems as it may contain values which are difficult to transfer - we will look at this problem in detail when we consider handshaking later.

UNICODE

ASCII is limited because there are only 127 codes available to represent characters. At present the result of this is that there are a wide range of ASCII derivatives, in which particular codes represent characters in a national character set, for example the £ sign in "UK" ASCII is held in the same place as the # character in the "US" ASCII set.

In order for documents to be completely international, and hold characters from all languages simultaneously, the UNICODE standard has been set up. This uses 16 bits to hold a single character, rather than 8. This increases the number of possible character codes from 127 to 65,535. The reason for this vast increase is that languages which use pictograms, for example Japanese and Chinese, have a very wide range of possible characters. UNICODE will become increasingly important as more software operates to its standard. It should remove problems encountered when moving documents from one country to another.

Windows NT and Windows 95 contain support for UNICODE filenames and future word processing applications will also use this.

Data Storage Devices

Main Memory

You can regard the memory space of a computer as a very long row of numbered boxes, each of which can contain a single item. The size of the box varies, depending on the computer system. On some systems a box can contain a number

of bytes, when more than one byte are stored at a particular location each location is referred to as a "word". The number of bits in a word varies, 16, 24 and 32 are common and it is nowadays rare to see a computer using a word size which is not a multiple of 8 bits. Computer memories in which each addressable location can contain a single byte are called "byte addressable", for obvious reasons.

How the data is stored in memory when the computer is working on it varies from one system to another, however when data moves out of the system, for example to be stored on a disk, it is almost always in the form of a stream of 8 bit bytes. In the next few weeks we are going to look at how the machinery of a computer system goes about storing and manipulating this data. Later on in the course we will look at the differences between the ways that particular types of computer system operate on data.

Magnetic Storage Systems

Magnetic storage involves making use of well established recording techniques, originally developed for sound recording, to store data. However because of the binary, on-off nature, of the data we are storing the media is optimised to store only one of two states.

Magnetic Tape

Magnetic tape storage makes use of well established recording techniques to store and retrieve information, although the media is not the same as that used for audio recording, being optimised for computer use. There are many different recording track patterns used, and many different tape standards.

Multi Track Tapes

Some tape standards involve recording more than one track on the tape. The "Industry standard" for computer recording uses 1/2 inch wide tape with 9 tracks down it, allowing a single byte and a parity bit to be recorded in one bit width. This form of tape can store up to 6250 bits per inch and, as it is available on 2400 foot reels can store a lot of data (the whole of the memory of a Sinclair Spectrum on about 8 inches). Other standards involve the tape being fitted into cartridges of one sort or another.

It is obviously impractical to store individual bytes on a tape when each byte takes up 6250th of an inch, particularly if you expect the tape drive to start and stop between each transfer. When data is stored it is therefore stored in BLOCKS, each block being separated by an inter block gap, which is the length of tape required to start and stop the tape drive. The size of the block varies depending on the application.

Helical Scan Devices

To increase the recording density of the tape, i.e. squeeze more data onto the surface, we are now seeing the use of helical scan tape devices. These make use of technology borrowed from video (Video 8) or audio (Digital Audio Tape) to store data. A helical scan tape machine uses tape heads mounted on a drum which is tilted at an angle to the tape. The drum rotates very quickly, causing the head to trace a number of narrow tracks up the tape. Because the head is moving very fast relative to the tape the recording bandwidth is very high, allowing large amounts of data to be written.

Densities of up to 2 Gigabytes are now common with such storage devices, although because of the recording density there are question marks over their reliability for long term storage.

Tape Access

Magnetic tape is known as a "serial" medium, in that you get to any particular piece of data by moving serially down the tape. Although most computer tape drives have a "fast forward" mechanism which allows them to quickly skip past unwanted blocks searching for the inter block gap they are still limited in the number of possible applications because of their serial nature. However they do provide a very cost effective way of storing large amounts of data, particularly for archive or backup purposes.

Tape Streaming

If a tape is being used to store individual files from a computer system it may well have to stop and start frequently, as each file is found on the disk and then written to tape. This inefficiency, which greatly reduces the speed at which information can be transferred, can be removed by using a technique called "tape streaming". If a disk is being backed up onto a streaming tape drive the system simply starts at the beginning of the disk and copies information onto tape as fast as possible. Files are not separated and what you end up with is an image on tape of the disk structure. This is not useful if you later want to pull a particular file from the tape, but is a much faster way of backing up the entire contents of a disk, being up to 10 times faster then a file by file backup.

Tape Backup

The most common use for a tape storage device is that of backing up hard disk. The usual policy is to use several generations of backup media, so that if one backup fails and damages the media, the previous version can be used. There are two forms of backup, full backup where everything on the system is saved, and incremental backup, where only files which have changed or been created since the last backup are written to tape. Most organisations take full backups very now and then and very frequent incremental ones. The standard policy is to make use of three backup sets; father, grandfather and son. These are rotated round so that at two copies of previous data are held while the current one is backed up.

Magnetic Disk

The magnetic disk works using the same recording technology as tape, but the magnetic coating is instead applied to a circular disk which is rotated. The heads which read and write the data are able to move across the disk and so can quickly access any part of data stored on it. Because it is much easier to locate and read any particular part of the disk this form of device is called random access.

The data is stored on the disk in a number of concentric tracks, each of which contains a number of sectors. Each sector equates with a block stored on magnetic tape, in that it stores a number of bytes (usually something like 256 or 512). Sectors are separated by a gap which allows the record/replay electronics within the disk drive (the machine which contains the mechanism to rotate the disk and the head assembly to read it) to switch on and off. The gap also means that slight changes in the rotational speed of the disk will not cause one sector to run over another if it is re-written.

Because the circumference of the disk is greatest at its edge some systems record a higher number of sectors on the tracks on the outer most sectors and then vary the speed of the disk depending on the position of the heads, so that the speed of the disk surface relative to the head is the same, regardless of the position of the head.

A similar technique is used to increase the playing time of compact disks. Unfortunately this technique reduces the speed at which data can be randomly accessed, as the drive has not only to move the head but also wait until the speed

of the disk has been set. If the drive controller can handle a wide range of data transfer rates however, you can transfer data more quickly from the outer tracks without needing to change the speed of the disk drive.

In some disk drives several disks are stacked above each other on the same spindle, each surface of which has its own head. This system, analogous to a number of tracks recorded down a single piece of magnetic tape, increases the amount of data which can be stored. Each individual disk of the assembly is called a platter.

Not all the surfaces of the platters are always used, for example if the disk is exchangeable the top and bottom surfaces will not be used to store data because they are too susceptible to physical damage.

You can calculate the amount of data it is possible for a disk to hold by using the following equation:

capacity = b x s x t x p

where:

b = no of bytes per sector
s = no of sectors per track
t = no of tracks per surface
p = no of surfaces per disk

Note that this just gives the raw data capacity of the disk, some of the space available will be required for directory information, of which more later.

"Floppy" Disks

"Floppy" disks are mainly used for data storage on microcomputer systems. Designed by IBM for loading system software into their mainframes they are now popular in applications where fairly small amounts of data need to be stored.

A floppy disk is a disk of magnetic media enclosed in a protective jacket which has a hole in it to allow the spindle to drive the disk around inside the jacket and a slot through which the disk surface is accessed by the disk drive read/write head. Originally released in 8 inch diameter format their size has progressively reduced, first to 5.25 inch disk which are now the industry standard, and now to 3.5 and 3 inch disks. The term "floppy" comes from the fact that the media is physically flexible, although the new 3.5 and 3 inch disks are encased in solid cases. The disk is exchangeable and the standard allows disks to be moved from drive to drive.

They work in exactly the way described above, with the head making physical contact with the disk. Usually the disk only spins when the disk is accessed, to reduce the wear on the disk and the heads.

They are available in two track patterns, 40 or 80 track, and two recording densities. The higher density (called "double density") uses more advanced head design and oxide coatings to increase the number of sectors in each track.

The disks can also be single or double sided and can be regarded by the host computer as one disk with two surfaces or two disks with one surface.

It is possible to record up to 800K bytes on a single disk, this may well double in the next few years as greater densities are perfected.

"Hard" Disks

Again, a hard disk works in exactly the same way as the disk described above, but it differs from a "floppy" disk in several ways. Firstly, as you might expect, the disk itself is rigid, usually made of aluminium with a very thin coating of magnetic oxide. It spins at a much greater speed than a floppy disk and the heads of a hard disk do not touch the surface of the disk. Instead they are shaped in the

form of an airfoil and "fly" in the current of moving air very close to the surface of the disk. Although they do not touch the recording surface they are sufficiently close to allow information to be recorded and replayed.

At no time to the heads of a hard disk actually touch the surface of the disk. If they do, in what is known as a "head crash", both head and disk are immediately destroyed by friction damage.

The heads are retracted when the disk is stationary and only loaded onto the disk by the disk drive when the disk has reached operating speed. Because they do not touch the disk they can be moved much more quickly about its surface, and the much higher rotational speed of the disk means that the latency is much lower than with floppy disks.

As you might expect, the system is very sensitive to dirt and dust. The distances between the head and the moving disk are so small that any dirt on the surface of the disk would almost certainly cause a head crash. The disk surface and heads are continuously "washed" in filtered air in order to stop contamination entering the drive.

The hard disk assembly, which usually contains more than one platter, is sometimes exchangeable.

"Winchester" Hard Disks

"Winchester" disks, named after the place where IBM designed the first one, are not exchangeable, i.e. the disk and the drive are one unit. Because the disk only has to work in the one drive, where conditions can be regarded as stable, more information can be packed onto the surface, for example the gap at the end of each sector to allow for different drive speeds when re recording sectors can be greatly reduced. Tolerances on track width can also be tightened up, allowing a much larger number of tracks.

Winchesters are not as sensitive to their environment as other hard disks, because they are assembled in clean rooms and then sealed, so that they are not susceptible to dust and dirt. Winchesters also differ from conventional hard disks in that the oxide coating on the disk surface is lubricated, allowing the heads to "land" on the disk if it stops spinning. This greatly simplifies the design of the head loading mechanism, making Winchester drives cheaper. The disk usually contains a special "parking" area where no data is recorded for the head to land and take off in.

Note that because a Winchester disk is a single, sealed, unit it is very difficult to recover information from a disk which fails in any way. This means that regular backup storage of information on the disk is highly important.

"Bernoulli" Disks

A new development, designed to combine the advantages of the exchangeable floppy disk and the high capacity Winchester disk is the "Bernoulli" drive.

This storage system makes use of a phenomenon discovered 200 years ago by the Swiss mathematician Daniel Bernoulli. What Bernoulli discovered is that if you spin a non-rigid disk very close to (but not touching) a hard, flat, surface the air pressure in the gap between the disk and the surface is reduced, stabilising the disk and drawing it closer to the rigid surface. This gives the "floppy" disk properties very similar to a Winchester and if you build the head assembly into the rigid surface you end up with essentially a "floppy" Winchester, although you cannot get quite the same packing density because your storage protocol has to allow for the exchangeable nature of the disk cartridges.

The disk portion of a Bernoulli drive is fitted in an exchangeable cartridge, to protect it from dust and dirt. Bernoulli cartridges can hold up to 20 Mbytes, about the size of a small Winchester drive.

The Exchangeable Winchester

A possible competitor with the Bernoulli drive is the exchangeable Winchester Disk. This is the moving, enclosed parts of a Winchester disk, including the head assembly but excluding the drive motors, mounted in an impact absorbing case. The idea is that you load this portion of the drive into a reader, which contains all the drive systems and electronics.

Optical Storage

Optical storage uses very fine laser beams to read and write information on media. Because a laser beam can be focused down to an area much smaller than we can make a magnetic read/write head you can store a lot more information using optical technology.

The system uses a media which contain tiny pits, or holes, which reflect or scatter a laser beam. The head positioning technology is much slower than that used for hard disk drives, because of the increased mass of the head and the need for much greater positional accuracy.

CD ROMS

This optical storage device makes use of the same technology as that employed by the audio industry in the Compact Disk. A modified CD player reads digital information from specially recorded disks. Because an audio signal is much more tolerant of errors than a digital storage device can afford to be great attention is paid to error checking and correcting when the information is read from the disk.

As the name suggests, a CD ROM is a Read Only device. But it can be very useful in applications where a large amount of data has to be distributed in as cheap a manner as possible, for example very large, fixed, databases etc. CD ROMS really score in that they can provide massive capacity, up to 650 Megabytes on a single disk.

CD ROM disks are now becoming a standard for the IBM PC. Many software companies make their programs available on CD, and this will increase in the future, because CDs are cheap to manufacture, currently hard to copy, and offer very high recording density.

The technology does however have limitations. The nature of the disk limits the maximum data rate which can be obtained off the surface and seek times are very slow compared with hard disks.

Most CDs sold nowadays are *double speed* which means that they can read data at around 3,000 Kilobytes per second. A standard CD can only transfer data at half this rate. They are also Multi-Session. Some CDs, for example those used with the Kodak Photo-CD system, are written to more than once. As successive sets of pictures are scanned and added to the disk a new track is created at the end. A Multi-Session drive is able to read beyond the end of one session, to find data for the next one.

The move is towards even higher transfer rates from the CD, with triple and quad speed drives appearing. The nature of the disk itself, and the way it deforms when spinning at high speed, is the ultimate limiting factor on CD transfer rate.

By the use of shorter wavelength "blue" laser diode technology the amount of data which can be stored on a disk may be doubled in the near future. Standards are now being finalised for double sided and multi-layer CDs with even greater capacity.

MPEG

The "Moving Picture Expert Group" (MPEG) has set a standard for data compression which allows up to 75 minutes of sound and colour video to be stored on a single CD. The compression used is "differential", in that frame difference information is stored and then applied to particular "key" frames.

The quality of the output is highly dependent on the encoding performed, which is a manually controlled process. Frames which contain large amounts of detail which is changing rapidly must be processed by hand to minimise pixellation and picture break up. It is not possible for PC users to perform their own MPEG encoding, instead they must decode pre-recorded data.

It is possible to obtain MPEG decoders for personal computers which allow such CDs to be viewed on the system. Future computer games may make use of this technology to increase the amount of video in a given game.

MPEG will have significant impact on the home entertainment market, which is currently based around video tape.

"WORM" Disks

A WORM (Write Once Read Mostly) optical disk uses a laser which can be operated at two power levels, one for reading and another, higher, level to actually burn a pit in the recording media. This gives the ability to write information on the disk, but once written the data cannot be erased, hence the name. The disk in a WORM drive is exchangeable, which means that you use it until you fill it up, repeatedly writing new versions of files which are updated. When the disk is full you copy all the valid information off in and then fit a new, empty disk. Since the data capacity of this technology is very high you would be able to use a disk for a very long time before it filled up, and because of the way it works the need for backups is removed entirely.

Some versions of the WORM technology employ disks which can be "wiped" clean by a thermal process.

Rewriteable Optical Disks

An erasable optical disk makes use of a magnetic technique to set patterns in the media when the laser is fired at a particular area. This allows you to treat an optical disk as a magnetic one, with two significant differences, the capacity is usually very great, and the speed is lower.

However, because the read/write head does not make physical contact with the disk surface it is easy to make exchangeable re-writeable media, so you can use this technology to provide large capacity exchangeable disks.

The SONY Mini-Disk, currently being promoted as a device for recording and playing back music, will also shortly be available as a computer peripheral. This will provide 150 Mbytes of exchangeable, rewritable, optical storage.

Managing the Storage Media

The above sections have described a number of devices which can be used to store information, in the form of 8 bit bytes, on a media of some kind. This section looks at how this raw capacity to store data is converted into a filing system by suitable software, i.e. how do we get from a machine which can be told to store this piece of information at track x sector y on a disk drive to a workable filing system.

Files and Directories

It is first worth considering the facilities which a filing system should provide before looking at how these are provided. All filing systems allow you to group information together in what are usually called "files".

A file is some information you wish to store. It could be a program, a piece of text etc. When you store information in a file on a computer you will give the file a name by which you and the computer will recognise it. As far as the computer is concerned a file is a string of bytes which you identify by a particular name.

In order to find the file you want the filing system keeps a list of the files currently stored and information about them, including where on the storage device they are kept. Other information which is usually stored includes the owner of the file, when it was created, when it was last changed, whether the file is currently being used by anybody, who has access to the file etc.

A list of files and this additional information is usually called a "directory". Information about a particular file is usually called the "directory entry" for that file. You do not need to worry about how the directory is stored and what is put in it, although the filing system will provide commands to find out what you have in the directory etc.

As far as the filing system is concerned, it is very convenient for it to regard a directory as a special kind of file. Several filing systems work like this, and therefore hold the interesting possibility of a directory which contains another directory and so on. Some filing systems, for example UNIX, VMS and MS-DOS permit this, and allow you to set up hierarchical filing structures. This facility is very useful, allowing related information to be stored in a particular directory.

Administering Filespace

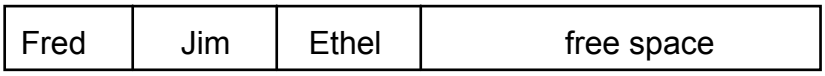
An operating system will use the directory entry for a particular file to find out where on the media that file is held. In this section we are going to look at techniques for allocating space on the media for a particular file.

In the case of magnetic tape the serial nature of the tape means that you simply record files serially down the tape. It is not possible to change the size of files once recorded, so the allocation system for magnetic tape is very simple.

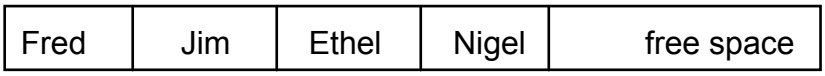
Random access devices, such as magnetic disks, allow us to be more adventurous. As we have seen above, the physical nature of the media means that the smallest unit of filestore which can be allocated is much larger than a byte, usually the size of a block (magnetic tape) or a sector (magnetic disk). The filing system is able to give the disk drive commands to put and get particular sectors. Assuming we have allocated a fixed number of sectors for our directory (a very simple minded filing system this!) we now have to decide how to give the filespace to the user.

A Simple Space Allocation System - The BBC DOS

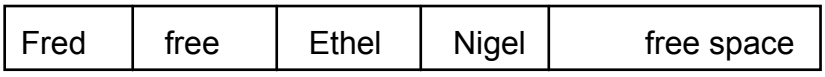
A very simple, and therefore quick, way of allocating space is just to grab the next free bit from the media each time the user wants to store a file, for example in the following:



space on the device is allocated from left to right, in this system we have three files: FRED, JIM and ETHEL stored on the disk. If a fourth file was added the system would simply allocate space from the free space and place the file there:



This is a very fast filing system, files are saved and loaded quickly and so far all seems well. However, consider what happens when we erase JIM:



The free space on the media is now split into two parts, which means that if we try to store a file which is larger than both of the free areas the system will tell us there is no room for it.

A second problem appears when we edit ETHEL and make the file larger. Because this would mean overwriting the next file along, NIGEL, the system must make a new version of ETHEL in the free area, using up more free space.

As time goes by the disk will become peppered with areas of free space which are too small to be useful, and eventually it will be impossible to add extra files because, although there may be enough free space it is not all in one place.

The BBC disk operating system works in this way, and provides the command *COMPACT to get around this problem. *COMPACT simply moves all the files down the disk, updating their directory entries and bringing all the free space back together. However, if a large filing system was implemented in this way we would find that such a technique would be very unworkable, particular as the whole filing system would need to be scanned at regular intervals, during which time all other work would have to stop.

A Slightly More Complex Version

In the simple system above files are stored in a contiguous areas on the media. A better approach is to store a file as a series of blocks. When the medium is empty all the blocks on it are available for use and as files are created they blocks are assigned to a particular file. If a file is deleted the blocks of which it is composed are returned to the filing system for use in other files. This gets around the problem of requiring contiguous space to hold a file in, as long as there are sufficient free blocks for a file it will be stored.

A very simple way of linking blocks together is to have a pointer at the beginning and end of each block pointing to the next block, and so on up to the end of the file. In fact it turns out that it is also a good idea to have a pointer to the previous block as well, so that programs can move in both directions within the file. This form of "linked" filestore is popular and is used on several operating systems. It is very resilient, in that if a portion of the disk is damaged it is usually possible to re-construct most of the files from blocks which are undamaged. The Cambridge TRIPOS operating system, which is the basis of the operating system used on the Commodore Amiga, uses a derivative of this system.

MS-DOS File Allocation Table

MS-DOS also uses a similar approach, with the refinement that a special area called the File Allocation Table is set aside on the disk which contains an entry for each block on the disk. Each file on the disk is allocated a unique number and that number is placed in the File Allocation Table entry for each block of the file. The links to each successive block of a file are also held in the FAT. When a disk is loaded MS-DOS usually reads the FAT into memory so that it can quickly decide which sector of a file is required.

Problems with "Blocked" Filestores

The main problem with a linked filestore is one of "fragmentation". Initially, when all the blocks are free files will be held in contiguous areas but as individual sectors are added and removed each file becomes spread over the surface of the disk. If the file is then accessed the disk heads will have to move about a lot to find the various parts. This can significantly degrade performance, with the filing system gradually slowing down as it gets older.

There is no way of easily solving this problem, which all operating systems which allocate files in chunks suffer from. The usual solution is to copy all the files onto another medium and then copy them back on to the disk in order. Some programs are available which can tidy up a fragmented system whilst it is running but the use of these can be fraught with danger...

A modern disk filing system - NTFS

The Microsoft Windows NT Operating System makes available a much improved filesystem. Because Windows NT is required to provide security and resilience (we will cover more of this later) the underlying filesystem must reflect this. Also limitations inherent in the FAT filing system make it difficult to use on very large storage systems.

The File Allocation Table system used by MS-DOS to manage storage is adequate for small storage systems, up to 1,000 Mbyte in size. However, it does have limitations in terms of maximum file size, limited filename length and very limited file attributes. It is also highly sensitive to damage, in that if the FAT is corrupted it is very difficult to re-construct the data. Also FAT directory entries do not support any form of security, i.e. anyone can access any files.

NTFS was designed to address the problems with FAT.

It supports file sizes of up to 16 Ebytes (2^{64} bytes). File names can be up to 253 characters long and files have access control lists attached to them.

The way that the files are organised on the disk is interesting. Rather than have a directory area and a file area NTFS has a large (4MB) system area where the directory items and attributes are held. This can be linked to a further area if the number of files on the disk becomes large enough.

The first few bytes of a file are held in its directory entry. This has two advantages; very small files take up virtually no disk space and once a file has been found some data can be supplied to the application instantly.

Directories are also organised differently. In an FAT system MS-DOS must search linearly through the list of files until it finds the required name. In the case of NTFS a B-tree structure is used to speed up file searches.

NTFS also supports transaction logging, in that writing to files is managed so that, in the event of a hardware failure, uncompleted transfers can be "rolled

back" to the point before they took place. Repair of disks damaged in this way should be virtually instantaneous.

Under NT all users have unique identities, and all resources have an owner, including files. The owner of a resource can give other users permissions to access that resource. NTFS file directory entries also contain an *Access Control List* which gives details of the users who are allowed to use a file and what the each user is allowed to do with it. There will be more detail on this area in the security section of the NT notes.

RAID Storage Systems

RAID technology, Redundant Array of Inexpensive Disks, arises out of the need for very large capacity, high reliability, storage systems. Even with a Mean Time Between Failure of 200,000 hours you still have problems with hard disk technology. If you had 100 disk drives, this would work out at around one failure a month. Furthermore a single, high capacity, disk drive will usually cost more than a number of smaller, cheaper, ones which can hold the same amount of data in total. Having lots of small disks can also improve performance, in that if they are on separate disk controllers they can be fetching different parts of the filestore simultaneously.

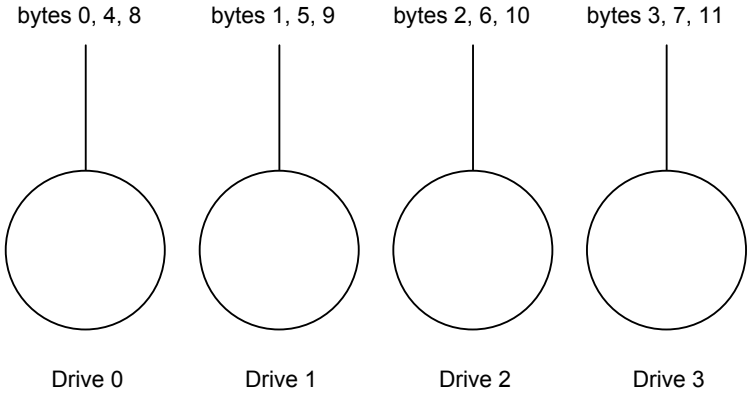
Various RAID levels have been set up, in each the array is used in a different way. Note that for RAID to work, the lower levels of the operating system; the portions which deal with file access, must be written to work in the required way. Note also that the way in which performance improves also depends on the way that the data is moved about; i.e. whether the system you are running tends to move large files about occasionally or lots of small files all the time. A final thing to remember is that RAID systems, because of the very large capacity storage you are creating, will have very heavy performance requirements, i.e. you will want to exceed the performance of a single drive whenever possible.

There are RAID levels to improve performance, reliability or both. Calling them RAID levels is a bit misleading, as the sequence is not progressive, i.e. RAID 4 is not based on RAID 3, you are better off thinking of them in terms of types or configurations.

RAID Level 0 - Striping

Striping is derived from the word "stripe", and pronounced accordingly!

Level 0 striping is a way of improving the rate at which you can pump data onto a disk drive. If you think of a single stream of data flowing onto a disk, it can accept the data at a particular rate, dependent on the speed the disk is revolving, and the performance of the controller. If you send the data to a pair of disks in parallel, you can get double the transfer rate, if you use 3 you can get treble the rate...



You send the first byte to the first drive, the second to the second drive and so on. For best results all the spindles are synchronised, so that the data lines up on the disk and writing takes place simultaneously. When reading data, it is sent off the drive in stripes as well, so you read it equally quickly.

However, there are problems with striping, particularly when you are reading and writing very small blocks. When using disks you have a particular sized unit which is the minimum which the disk can write at one time. This is usually referred to as a sector. If you have four disks in parallel as above, the smallest unit which you can use is four times this, 512 bytes. This means that a one byte file will take 512 bytes, wasteful of space.

You might decide to reduce the size of your sector (some disk drives can do this kind of thing) or you might decide to organise your file system with smaller clusters on each drive. Either way you will lose disk space because the smaller your cluster or sector, the greater the amount of space lost in sector and cluster headers etc.

The performance also suffers when you write very small amounts of data. If I am changing one byte on the a disk I need to read a whole sector, change the appropriate byte, and then write it back. On a single disk system, if I want to change two consecutive bytes they are usually in the same sector, so the amount of work the disk drive has to do is the same. On the striped system above if I wanted to change bytes 2 and 3 I need to read from two drives, and then write to those two drives.

The *striping factor* refers to the size of the blocks. In the example above, the striping factor is 1, i.e. one byte per drive. If I increase the striping factor I can sometimes handle the update of a sequence of bytes on a single drive, for example if my striping factor was four I could update bytes 1, 2 and 3 by just accessing drive 1. Note that increasing the striping factor will reduce the speed increase that I am going to get.

Striping does not improve reliability of course, if a drive in your stripe set falls over you will still lose data.

RAID Level 1 - Mirroring

Mirroring is much simpler than striping. It just involves duplicating the disk hardware so that reads and writes take place on two disks at once. This means that if one of the disk drives fails no data is lost, and the system can continue in operation on the remaining copy. The speed of data writing is not changed, but reading may be quicker, because you can access different portions of the data set at the same time.

Mirroring is expensive, but used where high reliability is of major importance. It is simple to implement, once the required hardware is in place.

RAID Level 2 - Error Correction

When considering computer memory, a number of techniques are available which allow defective memory components to be detected and the damaged data repaired. Such techniques, also called Hamming Codes, can also be applied to mass data storage. The data is spread across a number of drives, and some additional drives are used to hold the error correction information, for example you might have 10 data drives and 4 error correction drives. (remember that we are using comparatively cheap data drives here - and we are very worried about large capacity and great reliability).

This system is cheaper than mirroring, and has the advantage that a number of drives could fail and we can still recover the data and continue. In the above system, two drives could fail with no data loss.

Reading from such a system is still quite fast, because you get all the advantages of RAID 0. However, writing is slowed down, particularly for lots of small transfers, because all the data disks need to be accessed during a write - this is to calculate the parity information which is stored on the error correction drives. For this reason RAID 2 is not particularly popular.

RAID Level 3 - Single Parity Disk

Rather than having lots of error correcting disks, you could instead have a single parity disk. You can use the parity to restore data on a failed disk; however you now have a problem in that every write you make must also involve updating the parity disk. Before each write you must also read all the other disks, so that you can compute the new parity value.

This system is much cheaper than mirroring, in that you only need one extra drive, no matter how many drives you actually use. However you should bear in mind that this system will not keep going if a disk fails, it merely allows you to reconstruct the data on the failed disk so that you can continue later. Also, if more than one disk in the system fails you are still in trouble.

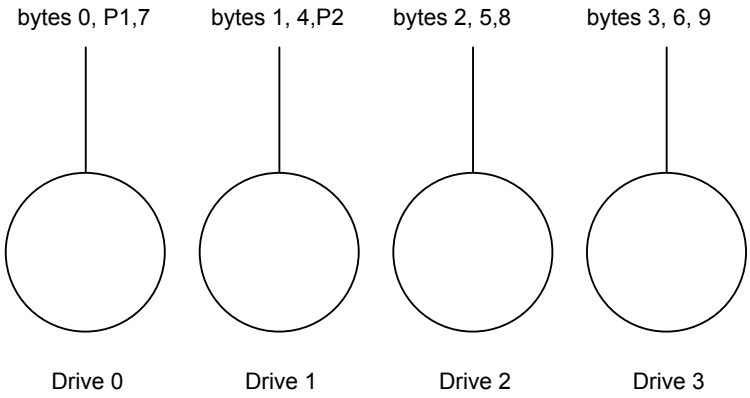
RAID Level 4 - Parity Disk with large stripes

RAID 3 has a low striping factor, so that most reads will involve fetching data from more than one disk. RAID 4 uses larger stripes, i.e. the striping factor is increased. The parity byte is then held for each stripe on the parity drive, for example if you striping factor was four, the first four bytes of the file would be held on drive 0, the second four on drive 1 and so on. The parity byte for the first four bytes would be the first byte on the parity drive, the second byte on the parity drive would be the parity byte for the first block on drive 1.

This improves read performance, in that you can often get all the data you want of a single disk. It also improves write performance, in that you only need to look at one drive to work out the parity byte. Note however that all writes still result in a write to the parity drive as well.

RAID Level 5 - Interleaved Parity

In the above parity systems the parity disk is an obvious bottleneck, in that all writes involve writing to this drive as well. RAID 5 is an improvement in that the parity information is spread on all the disks in turn :



P1 is the parity for the first four bytes, P2 is the parity for the next four bytes and so on. You still need to do two writes for each write, but the parity is not concentrated on a single drive.

Other RAID levels

There are other RAID levels available, 5+, 6 and 7. This differ mainly in the way in which the hardware is configured to support RAID.

Generally speaking, RAID can be used to improve the performance of hard disk systems, particularly where large files are used. If smaller files are accessed, RAID can be worse in performance terms than a single disk, but RAID technology does offer other benefits, in particular greater reliability.

Information Transmission

Serial Data

A serial data transmission takes place over a single channel, which the bits of data being sent one after another. The data sent down is either clocked at each end of the channel, or an additional clock signal is sent down a channel as well.

We will be looking at serial transmission systems which send data in bytes down a channel. Many processors, for example the Hitachi 63XX series, now have a serial port or UART (Universal Asynchronous Receiver/Transmitter) on chip.

RS Serial Data Standards

There are a number of "standards" for serial data transmission. In the USA the Electrical Industries Association (EIA) has laid down a number of definitions for serial channels. This standard is similar in meaning to the CCITT standard V.24. Serial connections to this standard are often referred to as RS232/V.24, although as we shall see calling these things standards is being somewhat generous!

RS 232

RS 232 refers to a standard whereby data is send serially down a channel in terms of a sequence of mark and space bits. The rate at which the data is sent is termed the baud rate (more of this later) and the number of bits in a character and the stop bits at the end of each byte also varies. (more of this later too)

In an RS232 connection the data varies between plus and minus 12 volts, with the two states officially changing at plus and minus 2.5 volts. This allows for a volt drop down a reasonable length of cable.

RS 423

A derivative of RS 232 is RS 423, which operates in the same way but works in terms of swings of plus or minus 5 volts, although at higher current. In theory this means that the data can be sent further, and should still be compatible with RS232 which works at plus or minus 2.5 volts.

However you must remember two things:

1. If you are running long cables between RS232 and RS423 devices the volt drop along the line may stop the RS232 end from detecting the data.
2. Many "standard" RS232 devices do not work down as far 2.5 volts, and can give problems even over short distances when talking to RS423.

RS 423 is becoming more popular, as only a single extra power supply rail, -5 volts, is required to implement the standard on TTL devices.

RS 422

Not as prevalent as the other standards RS 422 defines a differential standard where no fixed levels are used. This is not immediately compatible with either of the other two standards but can be made so by use of a buffer. Level Converters

The business of generating the voltage levels has been eased by voltage converters which contain inverters to produce the signal levels, along with buffers to drive the lines. These can be used to very good effect if you are building an RS232 interface onto a system.

Signal Paths

The RS232 standard, but not the V.24, defines a type of 25 way "D" type plug which is used to make the connections. The plug has two rows of pins, one with 12 pins in and one with 13 pins.

Although the standard gives a purpose for all 25 pins usually only a few are actually used, in fact other makers have made use of connectors with fewer pins, for example the BBC 5 pin domino plug and the IBM PC/AT 9 pin "D" type.

When the RS232 standard was set up it defined the serial connections as between computers, terminal equipment and modems in the following sequence:

DTE		DCE		DCE		DTE
terminal	RS232	modem	phone line	modem	RS232	computer
DTE		DCE		DCE		DTE

The RS232 data always flows between DTE and DCE equipment. If two computers, i.e. DTE devices, need to be connected you need a cable called a "null modem", which takes the place of the chain above. More on this later.

With the proliferation of computers, computers which can become terminals, printers etc. it is now hard to define whether a device is a DTE or a DCE. The only sure fire way to find out is to use a BREAKOUT BOX (see later).

DTE - Data Terminal Equipment

Usually the computers and terminals were defined as Data Terminal Equipment. By rights the DCE should have a plug on it, but this may not mean anything!
DCE - Data Communications Equipment

The modem was defined as the Data Communications Equipment. By rights the DCE should have a socket on it but see above!

Pin Assignments

The RS232 standard has names for the signals carried on each pin of the plug. Note that these names have different functions on a DCE from a DTE, although the documentation for a particular piece of equipment may not make it clear if it is a DTE or DCE, it will just give the names of the signals!

Note that we will name the data signals from the DTEs point of view.

Data Signals

2 TD - Transmit Data - serial data from DTE to DCE

3 RD - Receive Data - serial data from DCE to DTE

Handshaking Signals

These signals are usually active high, i.e. when the signal is in the mark state it is OK. There is a handshaking signal for both send and receive, and these lines are moved up and down by the communicating devices to start and stop transmission.

If these signal lines are used the handshaking is referred to as "hardware". If software handshaking is used (also referred to as XON/XOFF) you can ignore these signals.

4 RTS - Request To Send - set high by the DTE when it is ready to receive data, i.e. it is requesting the DCE to talk

5 CTS - Clear To Send - the DTE will only send data when this line is pulled high by the DCE

Note that other pins may be used to transmit what is effectively handshaking information, by using the equipment ready signals below, e.g. many printers pull pin 20, DTR, low when their buffer becomes full. This should be used in the same way as RTS.

Equipment Ready Signals

These lines used to be used by modems to indicate that all is well for data to be sent, in terms of equipment being switched on, a carrier being detected and the modem being ready to receive data. They are rarely used for this purpose nowadays, but many pieces of equipment require some of these signals to be set high before they will transmit.

You normally do this by looping some lines to a pin which you know is high in the connector.

6 DSR - Data Set Ready - set high by the DCE when it is active, but this may not mean that it is ready to receive data - only that it is operating. In the modem days this usually meant that the modem was powered up.

8 DCD - Data Carrier Detect - set high by the DCE (which would be a modem) when it detects a carrier signal from the distant modem, i.e. it indicates that a connection has been set up.

20 DTR - Data Terminal Ready - set high by the DTE (a computer or terminal) to indicate that it is powered up and ready. A modem will use this signal to determine whether or not to send to the terminal or computer, if this line is low it may remove the carrier from the connection, and so cause the DCD signal at the distant modem to drop, stopping information from being sent. This pin is very popular with devices like printers, for indicating when the printer is in a position to print.

22 RI - Ring Indicator - Used by an actual modem to indicate that a call is coming in. This line is pulled high by a modem to attract the attention of the host. I have never seen it used on printers and the like, but some software on the IBM is capable of reacting to this line by way of an interrupt and then responding to an incoming call.

Signal Grounds

1 FG - Frame Ground - this ground is the screen ground for the cable and the connector shields. DO NOT USE IT FOR THE SIGNAL GROUND. It is very rarely used, except in very electrically noisy environments.

7 SG - Signal Ground - this is the signal ground, always connect pin 7 at each end.

Weird Pin Assignments

Most of the other pins in the 25 line standard are never used unless you really are connecting a computer to a modem, in which case some of the clocking signals may be active. It is possible that some of the auxiliary lines may also be used, but this is not likely.

Since nobody uses the other pins much some manufacturers have used these other pins for other purposes:

- It is not unknown for a printer to be partially configured, for example to select between software and hardware handshaking, by the use of links between particular pins.
- In devices like multiplexors, where a single RS232 connection is "fanned out" to several, one of the pins is sometimes used to clock around the selected ports, i.e. a line is toggled up and down to move between ports. This system is usually used in cahoots with an input pin which is pulled high by the multiplexor when a particular port is selected. This gives the clocking device a way of establishing a start position.
- Most pieces of equipment pull a line high or low permanently, for use in setting connections at the far end.

The Null Modem

As I have mentioned before, if you wish to connect two DTEs or two DCEs you will need a "swapping cable" which takes the place of the modem path between them. Such a cable is sometimes called a Null Modem, because it appears to be a modem to both sides. The standard connection for a null modem is as follows:

1	----------	1
2	----------3	
3	----------2	
4	----------5	
5	----------4	
6,20	----------8	
7	----------7	
8	----------6,20	

Note that even if your equipment claims to be a DTE or DCE it may still use pin 20 for handshaking, in which case this connection will not work.

Data Rates

Once you have got your connection correct you are still not out of the woods yet, because the two machines must also agree on data rate, which is often referred to as the BAUD rate (E. Baudot or Binary Asynchronous Unit of Data). The baud rate defines how many bits per second we send, both machines must agree on this, otherwise nothing meaningful happens.

They are selected by tweaking dividers in the UART hardware or changing the values in timing loops or hardware counters.

If you are talking slowly to a thing which is going very fast it will read just one of your bits as a whole character. Consequently you will get lots of ^ characters cropping up. Note that if you get some characters coming through OK and others missing or damaged, you have got the correct baud rate but you need to adjust parity and stop bits - see later.

Split Baud Rates

Some machines can transmit and receive on different baud rates. This can be useful if you want to use the V.23 modem standard (see later) in which you send (i.e. type) at 75 baud and receive your replies at 1200 baud. The IBM PC is not supposed to be capable of doing this, having but a single clock for send and receive, although some clever programs claim to be able to do it.

Auto Baud Rate Select

Some communication ports claim to select their baud rate automatically, by trying different baud rates on incoming characters until they find character they "recognise" (either it matches one they are looking for or they do not get any parity or framing errors). These systems can be quite effective, but can go wrong if they happen to get something which makes sense at the wrong baud rate.

"Word" Length

The number of bits in a character can vary, usually it is either seven or eight, although there are some weird codes out there with five bit characters (ignore them!).

In order for things to work properly you have to know the number of bits in each byte, although if you are receiving you can usually get away with reading eight bit data and then throwing away the highest bit.

If the word length is seven it sometimes has a "parity" bit as the eighth bit (see later).

Stop Bits

Stop bits are sent after the end of the seven or eighth bit, to mark the end of the character. You usually talk in terms of one, one and a half or two stop bits. It is important to agree on the number of stop bits, otherwise you will regard an extra stop bit as the first bit of another character.

Parity

Parity makes use of an additional data bit which is sent at the end of each byte. This bit is set to a value which enables an odd number of bit errors in the data transmission to be detected.

The parity bit does this by being set or clear to ensure that the number of set bits in the data is even (even parity) or odd (odd parity). The receiver, which agrees with the sender on which parity is being used, looks at the number of set bits and flags an error when there is a discrepancy. Note that parity does not imply any correction, it is up to the controller to decide what to do when a parity error is detected.

Parity comes into its own when sending data down things like telephone networks, when going from one piece of equipment to another in the same room it just causes problems.

Duplex

Duplex is used to describe way that data is passed between devices down the communication channels. It again has its origins in modems, but is still used today to describe channels and connections.

Simplex

In simplex communication data is passed one way down a single channel. You could regard a printer as being connected to a simplex channel, unless it used

software handshaking, in which case a second channel would be required for the data from the printer to stop the host.

Half Duplex

In Half Duplex data is passed only in one direction at a time, this may mean using only a single channel, but in both directions. This usually involves the local terminal performing its own echo of characters sent.

Full Duplex

In Full Duplex characters can be transmitted in both directions at the same time over two channels. Just about everybody these days uses Full Duplex operation.

Software Handshaking (XON/XOFF flow control)

When the receiver is full of data it sends an XOFF (ASCII control character DC3 - 0x13), which the host reacts to by shutting up. When the receiver is again able to receive data it sends an XON (ASCII control character DC1 - 0x11), allowing the host to start talking again.

Software handshaking is popular because it reduces the number of connections required. However this protocol can cause great fun if you are operating over a noisy network and XOFF (and particularly XON) characters start getting lost.

Modems

A modem gets its name from the term Modulator- DEModulator. The modem takes a data signal and converts it into a form suitable for transmission down a particular media. One very popular use of modems is to convert data into audible tones for transmission down standard telephone lines, but this is not the only kind of modem. Another example is one which converts data into a signal which can be transmitted down a channel in a broad band communications network. These days the modem is much more complex than a simple transmitter and receiver. Modems often contain on-board processors and can perform data checking and compression in order to increase the speed and reliability of the data transmission.

They communicate between two ends, the Originate End and the Answer End. Which end you think you are affects the communication frequencies you use. Note that for things to work one end must be an originate and the other must be an answer. We will now look at some of the modem standards for use over standard British Telecom lines:

e.g. V.21 standard:

ORIGINATE		ANSWER
send 1180	---------->	receive 1180
980		980
receive 1850	<----------	send 1850
1650		1650

A remote user normally has an ORIGINATE ONLY modem, which allows him or her to call the ANSWER modem at the central site. Note that if two people have ORIGINATE modems they cannot communicate.

Most modems available these days can use both frequencies.

Hayes Compatible

An American manufacture devised a standard for computer control of an intelligent modem which can also dial calls. It has become very popular.

Auto Answer

An auto-answer modem will respond to a call coming in and alert the computer connected to it. This allows you to set up automatic data transfer to another machine, possibly using the cheaper out of office hours phone rates.

Portable

Portable modems are powered by a small battery, or from the comms port of the host computer. These are used by people who wish to contact the computer back at head office, for example salesmen etc. Some of the new portable computers also have modems built in.

Acoustically Coupled

All data modems need to be connected to the telephone network in some way. There are essentially two ways of doing this, direct connection and acoustic coupling.

A directly connected modem plugs directly into a telephone connection, usually using the now standard flat plug. This provides the best connection.

An acoustically coupled modem works by placing a small speaker and microphone assembly over the telephone handset. Because there is no electrical connection any phone can be so connected, and modems made to work like this do not need electrical approval. However acoustic coupling does not provide the best connection, modems working like this are more prone to data error and cannot operate at particularly high speed.

Modem standards

Standards have been set for data transfer between modems. These govern the frequencies to be used for the communication, and any negotiation the modems may take part in to set up a channel

V.21. - BELL - CCITT tones

The above V.21 example uses the audio frequencies given, and operates at 300 baud. However there is another 300 baud standard, popular in the USA and Japan, called BELL 103. Although the baud rates are compatible you cannot communicate, because the audio tones are different. It is possible to buy switch able modems however.

V.23 1200/75 baud

As an ORIGINATEr (i.e. sitting at home making calls to the host) you send data to the host at 75 baud and receive at 1200. Used for PRESTEL.

V.22 1200/1200 baud half duplex

Whilst popular in the states not used much in this country. BT do not like it!

Higher Speeds

The modems we described above makes use of different frequencies to demote 0 or 1. However, this approach is limited in terms of the data rate available, since the bandwidth of a telephone is limited.

If varying levels of amplitude and phase are used the number of data bits sent for a single baud can be increased, and this is how higher speed modems work. A number of points are available, each representing a particular phase and amplitude. Each of the points represents a particular bit pattern.

V.29 9600

This modem speed makes use of 8 different phase angles for 2 different frequencies giving 16 different events. Sixteen events is the same as four bits of data. The baud rate is 2400, so the overall data rate of V.29 is 4*2400 = 9600 baud

V.32

This standard uses a technique called *trellis coded modulation* to represent the different phase/amplitude settings. 32 *constellation points* are used, giving 5 bits per baud, however, one of the bits is used as an aid to error checking, giving a final data rate of 4*2400 = 9600.

V.32 bis 14,400

By increasing the number of phase and amplitude options the number of bits To improve reliability an extra bit can be added to aid in data checking. V.32 bis uses 128 data points, giving 7 bits worth of data per baud. One of the bits is used for checking, leaving 6 bits per baud and an effective data rate of 14,400

V.Fast and V.34

An improved V.32 standard, V.34 or V.Fast is becoming available which doubles the data rate of V.32, giving 28.800 bits per second.

V.Fast has been around in a non-standard form for a while. However the formal V.34 standard is now in place. Unfortunately, V.Fast modems will not necessarily be compatible with V.34 ones.

V.42bis and MNP 5

These are now compression techniques which are now part of the data transfer process. These involve the modem looking for repeated sequences in the data stream and representing the sequence with a single value when the same sequence is sent again. In the case of V.42bis the sequence is 32 bytes long. Such sequences, if repeated, are sent as a single value, giving significant speedup. The performance of this form of compression is data dependent, raw graphic images and text being very amenable to it. Program files and those which have already been compressed will however transfer less efficiently.

Parallel Data

In parallel data one bit at a time is sent down the communications channel. This allows us higher speed, at the expense of a greater number of connections. It is very unlikely that you would want to send parallel data more than a few yards, my rule of thumb is that parallel data is used to link systems in the same room.

Parallel data is usually sent in terms of 5 volt, TTL levels, which limits the distance the data can travel.

IEEE 488 - Hewlett Packard GPIB

This is a general purpose parallel interface developed by HP and enshrined in a standard by the IEEE. It provides a way of linking up to 15 devices on a network, with each device being either a talker, listener or both. Stations can command each other to talk or listen, with the facility of broadcasting to all stations on the network at once. Data is sent on an 8 bit wide data path.

You usually use IEEE 488 by designating one machine to be the master of the network, and every other device to be a slave; although it can be used as a general purpose network as well.

Hewlett Packard provide IEEE 488 connections on many of their laboratory equipment. Commodore used it for the disk drive and peripheral connections to and from some of their microcomputers, notably the PET series.

You can obtain chip sets and expansion cards which implement this protocol in hardware and, if you are lucky, provide an easy software interface to the protocol.

CENTRONICS Printer Interface

Devised by the Centronics printer corporation, this has now become the defacto standard for printer connection. Eight data bits are sent, along with handshaking signals which allow the printer to tell the host when it is ready for the next character. Timing diagrams for this printer protocol are usually available in the printer manual.

This interface is strictly for connecting printers to computers, i.e. data travels from the computer to the printer.

If you want to send a lot of data to your printer, for example if you are using a laser printer to do graphic images, it is best to connect the printer using Centronics, rather than RS232 because the data rate down a parallel line can be much higher.

Small Computer System Interface - SCSI

SCSI is a parallel bus system which is used to link processor and peripherals. Data is transferred in the form of addressed block; each device on the bus has a unique address and can send data to any other. Up to 16 devices can share a single SCSI interface. One is the master, and sends commands to the other devices. Each SCSI decide contains sufficient intelligence to fetch the data from whatever storage mechanism it is using and send it onto the bus.

SCSI is a very popular standard on workstations. It is primarily used for connecting mass storage devices such as disk drives and tape streamers, although it can be used for data capture devices and printers if required.

The SCSI standard is well defined, allowing SCSI devices to be plugged together very easily. However, in order to make use of a SCSI device appropriate driver software will be required for your machine, this may prove more difficult to get hold of.

A new standard, SCSI II is now becoming popular. It allows greater transfer rates.

Printers

In this section we look at some of the more common output devices used on workstations and consider their potential applications:

Dot Matrix Printers

Dot matrix printers form their characters out of dots. In this respect the quality of their output is lower than fully formed printing, where the character is printed as a whole. However, by reducing the size of the dots, and using many more of them, a dot matrix printer can today produce very high quality output. The great advantage of the dot matrix printer is that it can draw any required output, including graphical data. It is for this reason that this form of output is becoming predominant.

The precision of a dot matrix printer is often given in the form of so many dots per inch, for example 180 dots per inch. Do not read to much into this, the implication is not that the printer can produce dots 180th of an inch across, more that the printer head can be moved in steps of 180th of an inch, which is quite different.

In this section we will look at the different forms of dot matrix printer and their suitability for some applications.

Nine Pin Impact Dot Matrix Printer

The lowest specification dot matrix printer is the nine pin one. The name refers to the fact that the print head contains nine pin wires arranged vertically. When the head is drawn across the paper the pins are pushed up against a ribbon, making a mark in the appropriate place.

Nine pin printers usually use eight pins for the characters with the bottom pin being used for underlining. This makes the quality of them rather poor, i.e. you can easily see the dots which make them up. This printing quality is often referred to as draft.

You can get much higher quality printing from a nine pin printer by using what is often called Near Letter Quality (NLQ) mode. When printing in this mode the print head makes two passes over each line, moving the paper very slightly between them. In this way the resolution and contrast of the output is vastly increased. In such a mode, with a good ribbon and a following wind, a humble nine pin dot matrix printer can produce very impressive quality, particularly if you subsequently photocopy the printout! The penalty that you pay for the quality is speed; dropping from around 200 characters per second to 40.

Printer makers usually express the speed of their printer in "number of characters per second". This is a carefully chosen quantity which makes the printer look good, but can have very little bearing on the actual speed you get! It ignores things like the time it takes the print head to go back to the start of each line, the time it takes the paper to move up one line, etc. It is therefore best to regard these speed values as the same as the size of a car's engine; they give you an idea of the kind of speed you can get, but you need a road trial to find out the real value! In the same way, if you are comparing printers just use a standard document and time each printer printing it!

Most dot matrix printers have a bewildering variety of different print modes and formats. The host computer activates them by sending what are called escape sequences. An escape sequence is the ASCII character ESC (code 27), followed by a number of command characters which adhere to a particular standard. When the printer sees the escape character it then obeys the following sequence, for example on an EPSON printer ESC followed by M means "print at 10 characters per inch".

There are quite a few different standards for printer escape sequences but by far the most popular is the EPSON FX-80 range. All dot matrix printers, and some laser printers, can be made to work to this standard, I would advise you not to contemplate buying any other kind. Some manufactures support a superset of the EPSON commands, i.e. they have invented their own commands you can give in

addition to the EPSON set. Such extra commands are only useful if your programs know how to use them, at this point we have to consider the fraught business of printer drivers.

Most user applications want to drive the printer. However there is a very wide range of printers, each with its own subtleties. Software manufactures get around this by interposing a printer driver between the application and the printer. The application uses a standard form to express the various printer facilities it wants to use and the printer driver for a particular printer then converts a request for a particular feature into the commands for the printer you are using. Things become even more complex when you realise that different printers often have entirely different characters sets and sizes, all of which must be handled by the host. Writing printer drivers is therefore a tricky business not for the faint hearted. It is best to make sure that the application that you buy has a driver for the printer you want to use. You can make this a virtual certainty by only picking printers which support standard commands. Do not be impressed by a smooth demonstration of wonderful extra options, these are only useful if your word processor can select them! When I consider each style of printer I will recommend a standard for escape sequences, for the nine pin printer that standard is EPSON FX-80.

Twenty Four Pin Impact Dot Matrix Printers

As you might expect, this form of printer has 24 pins in its print head rather than nine. These are not normally aligned vertically, as there is a limit to how small you can make a print wire. Instead they are staggered.

The print process is exactly the same, but note that although we have more than twice as many pins we do not have double the resolution, because the pins are not actually half the size. This form of printer can print Near Letter Quality in a single pass, which results in faster high quality output. It can also print in draft mode much more quickly as well. However it has been my experience that the main benefits of a Twenty Four pin printer are in terms of speed, nine pin printers can get close to their levels of quality. These printers can be more expensive to run than nine pin ones, the ribbons cost more because they have to be made of a finer weave material, which the smaller pins do not punch holes in.

The standards for 24 pin printers are less well established, the best ones to go for are EPSON LQ compatible.

Ink Jet Dot Matrix Printers

These printers make marks on the paper by squirting ink at them. The ink is held in a reservoir behind the print head and forced in some way, either using a piezo-electric crystal to shake it out of the nozzle or by heating the ink so that a bubble of steam forces it out. It is then attracted to the paper by a high voltage potential difference. The big problem with such printers is that the ink tends to soak into the paper. You often have to use specially coated paper to get the best results, this can be expensive. However, developments in ink technology mean that the Ink Jet can now produce output on plain paper which rivals the quality of laser printers, at a much lower cost. Problems with blocking ink delivery systems have been solved by making the print head and the ink reservoir a single component.

Ink jet printers also handle colour very well by having four print heads, on for each primary and one for black.

The leaders at the moment in ink jet technology are Hewlett Packard with their DeskJet series and Canon with their Bubble Jet devices.

Laser Dot Matrix Printers

A laser printer is a descendant of the photocopier. A photocopier works by focusing the image of the source onto a selenium drum. This builds up a charge pattern on the drum which is used to pick up fine powdered ink. The ink, called toner, is then heat bonded onto the paper, forming the required image. A laser printer forms an image on the drum by driving a semiconductor laser under computer control.

The laser can "paint" on the drum to a particular resolution, which is usually 300 dots per inch. In this respect a laser printer is really a glorified dot matrix printer, albeit with very high resolution.

The laser printer engine must print the whole page at once, so the image must be completely formed in memory. If you do your sums you find that an A4 page at 300 dots per inch is several megabytes. Filling this needs a processor with plenty of oomph, 68000s are popular with RISC chips and transputers coming along. Two things come out of this, laser printers can vary greatly in speed (particularly on complex graphics) and you should make sure that the printer has enough memory, 2.5 Mbytes is around enough.

Laser printer speeds tend to come in two flavours, the engine speed i.e. the rate at which a page is printed and the time it takes to load the data. The rates are usually given for printing at speed, the machine often take a lot longer to print the first page.

Laser printers are actually based on only a few mechanisms, with ones by Canon and Ricoh proving very popular.

Some printers get their image on to the drum without using a laser, liquid crystal shutters and very small leds are also used, but the resolution is generally the same.

300 dots per inch is OK for office work, although you can see the edges of some characters. We are just seeing 400 dpi printers appearing on the market. These should compare better with 1000 dpi phototypesetters.

The current standard for driving a laser printer are the Hewlett Packard LaserJet II and LaserJet Plus ones, most printers will support these. More expensive ones can also be persuaded to perform as Epson dot matrix printers, daisywheel printers and even plotters.

The laser printer, and like devices, can give us a very high quality printout. However, with such a powerful printing engine you must consider how you are going to get it to make the marks on the paper that you want. When we looked at Dot Matrix Printers we found that one way to take complete control of the output was to drive the printer in bit image mode. This means that the computer software has to give information for each dot the printer makes on the paper.

Laser Printer Control Languages

The HP Laser Jet printer language is designed to drive a 300 dot per inch printing engine. All drawing operations are tailored to this resolution and fonts are supplied as raster images to this resolution. This means that different sized character must be specified individually, as different bit images. They are usually held in ROM cards plugged into the printer, or loaded into the memory of the printer before a print job.

The printer is simply capable of executing print commands supplied to it from the host computer. Tasks such as image transformation and scaling must be performed by the host and the resulting large bit image sent to the printer.

Postscript

The Adobe Postscript Page Description language is a programming language specifically designed to allow printed output to be specified. It is not tailored to any output device or resolution in particular. Fonts are specified as outlines, i.e. a series of mathematical specifications of lines, curves and points. These can be rotated and scaled by the printer itself performing the required mathematical transformations. This means that a single font outline can be displayed in any required size, with no loss of detail.

Complete programs can be loaded into the printer, which is then capable of executing them, obeying commands which result in output on the printed page. The printer is able to perform image transformation and scaling on data supplied. This reduces both the load on the host machine and also the amount of data which needs to be transferred to the printer.

The Postscript language is stack based. This means that all operation takes place around a central stack. Postscript programs push things onto the stack and then use operators to manipulate them. In terms of programming, Postscript is perhaps closest to FORTH, with the important difference that, in Postscript, data of any type can be pushed onto the stack with a single operation. It is quite acceptable, for example, to push an entire font outline onto the stack and then invoke a Postscript word to perform a particular scaling transformation on that font. Such a word would return the scaled font, again on the stack. The actual graphics portion of Postscript is reminiscent of the LOGO programming language, in which you drive a turtle around the paper, giving movement instructions.

Postscript also has mechanisms whereby you can specify bit mapped images as well as drawings made up from vectors. These images can be transformed and clipped by operations in just the same way as any other.

One of the drawbacks of Postscript is that a Postscript printer will not just print text out if you send some to it. It needs to have a program which contains instructions to print the text required.

Thermal Wax Dot Matrix Printers

The laser printer represents state of the art for most office printing jobs, but is not able to produce very high quality output because of its limited resolution and the fact that the resolution is supplied as either on or off. If you want to print out grey scale pictures you must use several dots to get a particular level of grey. This results in a drop in resolution. Because of the difficulty with grey scales colour laser printers are not yet popular, they tend to simply three or four laser printers in one, printing Cyan, Magenta, Yellow and Black.

One way of printing high quality colour is by the use of a thermal wax printer. These operate by heating a plastic membrane which is coated with a wax based ink. The ink sublimates, i.e. turns into a gas, and ends up on the paper in the form of a coloured dot. By varying the amount of heating you can control the amount of ink which is taken off the "ribbon", and so get different amounts of colour. This form of printing is very expensive, as four sheets of ribbon are required for each page, but it is the only way of getting high quality colour output onto paper at the moment.

Daisy Wheel Printers

The daisy wheel printer is currently declining in popularity because of the rise of the laser printer. It works in a very similar way to a typewriter, using fully formed characters which are pressed against a ribbon to mark the paper.

In the case of the daisy wheel printer each character is stuck on the end of a plastic "petal". Each petal sticks out of a central hub, which is fitted on to a

motor inside the printer. To print a particular petal the motor spins the "daisy" to bring the required character form up in line with a solenoid which then presses that petal against the ribbon, making the paper.

The character quality is very high, because the letter is "fully formed". However a major limitation is the fact that only the characters actually on the daisy can be printed, additional type styles such as italic cannot be printed without changing the daisy. Because of the way in which the printer works, it is difficult to make it print more than around 30 characters per second, and this printing method is also very noisy.

Until the laser printer came along daisy wheel printers were very popular. Two companies, Diablo and Qume, became very rich making and selling them. The most popular standard for such printers is Diablo 630.

MS-DOS

What is an Operating System

An Operating system is the piece of software which you add to the hardware of a computer to make it useful. It is closely coupled to the hardware and is usually supplied by the computer manufacturer. Operating systems which you may have used include VOS (Vulcan Operating System for the Harris), VMS (Virtual Management System for DEC machines), UNIX (operating system for several different systems.

Whenever a computer is being used, even if it is running a program written by a user, some part of the operating system is still present, looking after low level activities which keep the computer going, for example handling user inputs and displaying output.

Features of Operating Systems

Virtual Memory

Some operating systems allow you to use a mass storage medium, e.g. a disk drive, as an adjunct to main memory. They provide programs running on the system with the illusion of a massive main address space and then translate attempts to use this into access to the appropriate device. We will look at Virtual Memory later.

Multi-Tasking

Some operating systems allow more than one task to be in operation at once. The do this by "time slicing", i.e. switching rapidly between each active process in turn, giving the illusion of several programs running at the same time.

Process Communication

If more than one process is allowed it is reasonable to allow then to communicate. Some operating systems allow pipes, or mailboxes to be set up by processes who wish to talk to one another.

Device Independence

When you write a program to run on a computer that program may wish at different times to talk to many different devices, for example for debugging you may wish to see the output on the screen but for the production run you may want to send things to a file. Device independence allows you to treat all the different devices as the same, the operating system then does the required work to handle the different mediums.

Hierarchical Filestore

So far the filestores that we have looked at are flat, in that all the files are held at one level. This is not a very useful way to organise your work. If you consider that a 20 Mbyte disk may have over 2,000 files on it you can see that a flat filestore would be very hard to use.

A hierarchical filestore allows you to build up a structure into which you place your files. Files with a particular purpose (for example all my PASCAL programs) can be held in one area. The structure is built up like a tree, with branches going off in different directions and sets of files held at each point on it.

Emulation

Some operating systems, for example Windows NT, make use of the power of the computer they are running on to perform operating system emulation. A particular piece of software is designed to work on a particular computer, running a certain operating system. For that software to work on any other system it must be modified. This can often cause people to put off moving from one operating system to another, because all their applications will stop working.

Emulation is a technique where the operating system is constructed so that it can put a shell around applications and make them think that they are running on a different system. The emulation is not usually very efficient, particularly if the host computer also has to emulate the processor as well as the operating system, but such devices are very useful when migrating from one type of operating system to another.

MS-DOS History

MS-DOS was created by Microsoft and was originally based on an 8 bit operating system similar to CP/M. It has progressed through various versions, we are currently at 3.3.

It was picked up by IBM as the operating system for their 8088 based PC. PC-DOS is available only from IBM. MS-DOS is virtually identical and can be bought for many different "clones" of the IBM.

MS-DOS is now a very old system, with layers of modifications to try and keep up with the hardware changes which have taken place. It is still the underlying system of most IBM compatible computers, even the Windows graphic environment is mounted on top of DOS. However, this will change with later versions of Windows and new operating systems like OS/2 and Windows NT.

MS-DOS facilities

MS-DOS is booted into a machine from hard or floppy disk (or over a network - see later). A special MS-DOS program called COMMAND.COM supports all the mainstream commands (internal). Other commands are handled by separate programs (most operating systems work this way).

MS-DOS Memory Map

MS-DOS is fitted into the memory of an IBM compatible in a particular way. We have 1 Mbyte of address space to go at. The layout is as follows:

ROM bootstrap
Other utility ROMS
Display Area
Transient COMMAND.COM
Transient Program Area
Resident part of COMMAND.COM
Installable drivers
File Control Blocks
Disk buffer Cache
DOS kernel
BIOS
Interrupt Vectors

Taking these in turn:

ROM Bootstrap

When the 8088 is powered up or reset it will go to the top of memory and start executing whatever it finds. Usually means test the system and then go to a disk and try to boot up.

The bootstrap also looks in the ROM utilities area to see if any additional device drivers have been fitted. If you add an expansion, for example a disk drive, it will contain a ROM in the additional device drivers area to be called to set things up when the machine starts.

Also contains resident parts of the BIOS, i.e. the very low level interface to the machine.

Other Utilities

Allows ROMS to drive other devices to be placed in the memory map, for example special interfaces, GPIB, Hard disks etc. It is looked for on power up, see above.

Display Area

This area of memory is reserved for the display. The display is "memory mapped", i.e. it is scanned by additional hardware which generates a video picture which depends on the state of the memory, e.g. in its simplest state a bit set in the memory lights a particular dot on the display.

The output is placed in the display area by the BIOS (see later) which may call the bootstrap ROM to do this.

Transient COMMAND.COM

Portion of the command processor which can be overwritten when running large programs. If this area is not present when a program finished it will be reloaded from a disk drive connected to the system.

Transient Program Area

Where everything happens. Programs can request chunks of this and can also stop without leaving it. Such programs are called TSR "Terminate and Stay Resident" and are popular as pop up utilities. Note that once terminated in this way such a program cannot be re-activated by MS-DOS and so must provide another means of activation, see interrupts below.

Note that if you add large TSR programs the amount of memory you have left to actually run things is reduced.

Resident COMMAND.COM

Asks you to re-load the main program if it finds that the transient portion has gone. It also handles BATCH files.

Installable Device Drivers

Can be linked into MS-DOS to allow a high level interface to other devices, e.g. networks, peripherals etc. MS-DOS specifies a form which these drivers must have. If you wish to connect something exotic to MS-DOS you can do it by writing a device driver to this specification. The advantage of doing this is that any other programs running under MS-DOS can then access the new device via your driver and they do not need to be changed to do this.

File Control Blocks

Space for file buffers, pointers etc. for files which may be opened. The size of this area limits the number of files which can be open at one time.

Disk Buffer Cache

Used to speed up access to a disk by keeping portions of a disk in RAM. May also hold FATs here.

DOS Kernel

Low level interface routines loaded at boot. They provide routines to manipulate the disk buffer cache and file control blocks to give a filing system. They also look after the directory entry and FATs for disks.

BIOS

Loaded from disk when the machine is booted. It handles low level input-output and interacts with the bootstrap ROM which usually does the work.

If you write programs in Assembler you will make calls to the BIOS to do your input/output as well as a number of other functions. If you use a compiler it will generate calls to the BIOS for you. BIOS routines are accessed via "interrupts" - see below.

Interrupt Vectors

Two functions for Interrupt Vectors, hard interrupts and soft interrupts. The vectors point to particular software which handles these interrupts. These vectors can be changed (with care!) to greatly modify what the machine does when a particular event happens.

Hardware Interrupts

When a hardware interrupt occurs, i.e. when a piece of equipment connected to the computer attracts its attention electrically, what actually happens depends on the contents of an interrupt vector. There is a separate vector for each different

type of hardware interrupt. Usually it points into the BIOS, which handles low level activities of this type. However, by changing the contents of the vector you can cause another piece of program to be obeyed when a particular interrupt occurs.

An example of this in action is the TSR programs mentioned earlier. Most of these re-direct the keyboard interrupt into themselves and then look at each key before passing key presses onto the BIOS. If a key is recognised by the TSR program (the key used to activate the program is usually called a "hot" key) the TSR program takes control. This allows a crude form of process switching, you can stop one program and start another by pressing the "hot" key. When you have finished using the TSR program it will return you to the main one, which does not notice that anything has happened. TSR programs are usually of the general purpose type, providing desk diary, notepad, calendar, calculator etc. facilities which can be accessed whilst using other programs.

You would also re-direct an interrupt vector if you wished to directly respond to a particular piece of hardware.

Software Interrupts

As well as the hardware interrupts described above a set of interrupt functions is also directed by a set of vectors at the base of memory. These functions are generated by internally generated interrupts, i.e. the 8086 and 8088 have machine code instructions to simulate a hardware interrupt. These are used in MS-DOS as a quick and easy way of calling a function routine, since an interrupt call is far shorter than a subroutine call.

These function interrupts are never used as the result of anything happening to hardware, they are used to request functions from the BIOS. MS-DOS reserves interrupts 20H to 3FH for this purpose. A popular interrupt is interrupt number 33, which will do all manner of things, depending on function requests passed in the registers when it is called.

Note that because command interrupts, along with all the others, can be re-directed, you can change command handling routines and make them do different things. A common use of this facility is to change the function of particular commands, i.e. look at the command code and respond to particular requests. This can allow you to mimic functions which a user may ask for in a different way, or stop a user from doing something.

MS-DOS in Operation

Having looked at the way in which the operating system is fitted into the machine we can now look at the operation of the system.

Starting up MS-DOS

This section gives the sequence of events which takes place when a computer system which runs MS-DOS is started up.

The Self Test sequence

This is not performed by MS-DOS, instead it is a machine specific routine which is part of the boot ROM. Before the computer tries to run MS-DOS it makes sure that all is well with the hardware. This includes such things as checking that all the external devices, e.g. printer, keyboard etc. are connected properly and that the memory is working correctly. If a piece of hardware fails the self test, e.g. the keyboard is not found - perhaps because someone has disconnected it, an error number is displayed which can be used to identify the problem.

All computers have some form of self test routine when started, in the simplest ones this just determines the amount of memory present, usually by the simple minded approach of writing up memory until something different comes back.

Note that just because a machine passes its self test does not mean that it is trouble free, thermal and flaky problems may not be detected reliably.

The Boot Process

Booting up a system, from the expression "pull yourself up by your bootstraps", happens when the machine is powered up or reset. MS-DOS is not stored in the hardware of a machine, instead a small piece of code called the "bootstrap loader" is held in ROM. The loader starts the machine by looking at a disk in an external drive; if it is a system disk it loads the "boot sectors" from the disk and calls them. This makes it very easy to run different operating systems, all you need is a different disk.

Note that system disks are specially identified. Not all of your floppy disks will be system ones, because this would waste up to 100K of a disk with files that are only needed when the machine starts. When you format a disk you are allowed to specify that it is to be a system disk; this causes the system files to be copied into the correct positions. Note that although the system files are copied the configuration information etc. is not, so you will have to add extra files (see below) before this is a useful disk.

The first section of the boot program takes control and brings in the rest of MS-DOS. Once loaded the operating system starts to configure itself to the hardware and the options that you have selected:

Configuration Information

Two files are used by MS-DOS to find out what devices and configuration you wish to use. They are specific to a particular machine and setup, and can be the cause of a program failing to run on two apparently identical machines. The two files are called CONFIG.SYS and AUTOEXEC.BAT.

Note that these file are used once only, at boot up. If you change either of them you will then have to re-boot the system for the changes to have effect.

CONFIG.SYS

This is a file of special information which MS-DOS reads as it is booting up. It tells how much space is to be reserved for disk I/O, information about the country where the machine is being used and also names any special device drivers that you may wish to use.

There are a wide range of configuration options which you can put in this file and the way in which it is used varies from one version of DOS to another. The latest version of DOS, 6.2 allows you to select different config.sys settings from a menu when your machine boots up. This can be useful if you need to run programs which require conflicting types of settings.

MS-DOS and Micro Processors

The IBM PC was developed many years ago, using the then quite advanced INTEL 8088 microprocessor. MS-DOS and PC-DOS are designed specifically for use on this chip, and the use of other, more advanced, processors has been restricted by this fact. Taking the processors in turn :

Intel 8088 and 8086

In the beginning there was the 8086. Intel developed it as a processor with 16 bit registers and a 20 bit address bus. The 8088 and 8086 are functionally equivalent, and are programmed in exactly the same way. Their difference is that the 8088 was designed to function with an 8 bit data bus, i.e. the 16 bit registers are loaded using two successive fetches from memory. This slows the machine down a little, but does allow the use of standard 8 bit components when you are designing the machine, making it cheaper and easier to build.

IBM chose the 8088 as the chip for the PC, and designed everything else around it.

Manufacturers who want to make PCs which go faster now use the 8086 in their machines, and multiplex the data bus when talking to 8 bit devices. This generally works OK, but can lead to problems when talking to strange pieces of hardware in the expansion slots.

There are two principle limitations of the 8086 family of processors :

- They only have a 20 bit address bus, making it impossible to address more than 1 Megabyte of main memory. MS-DOS took this limitation on board, leading to a maximum program size of just 640K. Today's programs need a lot more memory that this, which leads to strange and inefficient memory enhancements, of which more later.
- The 8086 only contains 16 bit registers, which means that it has to use trickery to handle the 20 bit address range. It does this by splitting an address into two chunks, a 16 bit offset and a 4 bit segment number. This allows the full range of memory to be addressed, but it also means that the largest chunk of memory which can be directly addressed is 64 kilobytes. This is in contrast to other 16 bit processors, for example the 68000 range from Motorola, which have 32 bit address registers and can regard memory as one continuous lump.

However, chip technology has moved on from the 1970's when the 8086 was designed and Intel have produced a number of upgrades on the original chip :

Intel 80186

This was the first upgrade of the 8086. It is an upgrade of the chip which does not greatly increase its power, but does make it slightly easier to construct circuits around it. Consequently not many PC compatible machines are based upon it, but you do find it cropping up in dedicated systems and sometimes as a slave processor on an expansion card (quite a few intelligent Network Interface Controller cards use this chip).

Intel 80286

The first significant upgrade of the 8086. Intel were aware of the dangers of moving away from the 8086, and so designed a processor which could operate in two modes. In one mode, "real" mode, the processor was nothing more than a souped up 8086, running at around 3 times the speed (depending on the clock speed). In the other mode, "protected", the 80286 allowed lots of extra goodies; up to 16 Mbytes of main memory, handling of paged memory, multitasking, etc. The problem was, no software was available to make use of this extra mode. IBM released their upgrade for the PC, called the PC/AT, but nobody made use of the "protected" mode abilities of the chip. To make matters worse, changing

from real to protected mode involves resetting the entire processor, not an easy thing for programmers to get their heads around.

The 80286 has been referred to as "brain damaged", in that this and other limitations of protected mode have stopped it from being used properly. The operating system, OS/2, which was designed to make full use of the 80286 has not met with great success and so this chip must be regarded as a bit of a dead end.

Intel 80386DX

This is the chip where Intel got it right. Again, it has the ability to operate as a fast 8086, but it also has a lot more besides. In addition to the "real" mode as an 8086 this chip also offers a number of additional modes, along with much improved methods of changing from one to another. The 80386 has a "protected mode", in which it functions as a processor with 32 bit registers and a 32 bit address space, allowing huge amounts of memory. It can also emulate a number of concurrently executing 8086 processors, and perform the memory mapping and memory access control to make this work. This means that it is the perfect processor for running multiple versions of original IBM PC applications. Software such as Windows 3 and Quarterdeck make use of this to allow DOS multi-tasking.

The memory mapping and access control allows ordinary MS-DOS programs to multi-task and also supports the use of disk storage as an extension to main memory. Because the 386 can be made to detect when a program access a particular region of memory, and then invoke an appropriate system handler, competing MS-DOS programs can be handled without extensive modifications to them.

Intel 80386SX

The 80386DX was the original version of the 80386 processor. It came on a chip which supported full 32 bit address and data buses and was consequently more expensive to build a computer around. Intel made it easier and cheaper to produce 80386 based systems by launching the 80386SX. Internally this chip is identical to the "full" 80386DX, but the connections it offers to the world are 24 bit address lines and 16 bit data lines, i.e. the same as the 80286. This means that computer makers can upgrade 80286 based machines using the 80386 and only need to make minimal changes to the design of their machines.

The 80386SX is to the 80386DX what the 8088 is to the 8086, a chip which is easier to build a system around, but with the same internal structure.

In use, the fact that the data bus is only 16 bit does not greatly affect performance, and this chip has become the basis of a new "standard" machine.

The biggest limitation of the 80386SX with respect to its larger brother is the limited address space. 24 bits only allows for 16 Megabytes of main memory, which may prove a limitation in the future. Looking at the problem another way, the extra functions of 80386SX machines makes saving a small amount of money by going for an 80286 based machine very misguided economy.

Intel 80486DX

The 80486, in contrast with the progression of the chips above, does not do anything that the 80386 does not, i.e. its advantages are in terms of raw speed. It combines the functions of an 80386 and an 80387 on a single chip, along with 64K bytes of internal cache memory. Cache memory is very fast memory directly coupled to the processor. Frequently accessed data is held here, to

reduce the requirement to access slower memory outside the chip. In addition the 80486 hardware fetches program instructions from memory and puts them into the cache before the processor needs them, so further increasing the speed.

The 80387 is a numeric co-processor. Each Intel processor in the above range has a complimentary numeric "co-processor". This sits across the same data and program buses as the main processor and performs floating point arithmetic in hardware, as opposed to the software routines which are normally used. This can speed up numeric applications dramatically. In previous versions of the chip the co-processor was separate, and most PC compatibles provide a socket on the processor board into which a co-processor can be added.

When making the 80486 Intel took advantage of techniques of even larger scale integration, and put the numeric chip on the same piece of silicon.

The 80486 doubles the speed of the 80386DX, and if you are considering a machine for a numeric application, when you were going to buy a numeric co-processor anyway, is well worth looking at. The price of 80486 based machines is currently dropping at speed.

Intel 80486SX

The 80486SX is **not** related to the 80486 in the same way that the 80386SX is related to the 80386. In the case of the 80386 the SX version is functionally identical, i.e. it will run all the same programs, only the way that it is connected to the system is different.

In the case of the 80486SX Intel have been sneaky and left something out of the 80486. The thing missing is the 80487 co-processor. This means that you can regard an 80486SX as an 80386 which goes fast, **not** a cheap version of an 80486. If you want to do things which need a co-processor, most CAD drawing programs and mathematical packages need them, you will have to buy an 80487SX chip.

Pentium

The latest Intel chip is called the “Pentium”, as opposed to the ‘586. It offers no additional operating modes or instructions, but has a 64 bit data bus, increasing the rate at which data can be transferred in and out of the processor.

It also makes use of dual processor cores, so that if the code is ordered correctly programs can be executed more quickly. Ordering the instructions in this way is normally performed by the compiler used to create the binary version of the program. Programs created in this way can run significantly faster on a Pentium, but this performance improvement is quite application specific.

DOS and Memory

The original PC, from which all current machines have descended, was able to address 1 Mbyte of memory. This amount of memory is deemed merely adequate for a display system now, but when the PC was first created such a large amount was thought to be somewhat excessive. This 1 Mbyte address space simply means that the processor, the chip which controls what the PC is doing, can uniquely address 1,000,000 separate locations. This sets an upper limit on the amount of program and data which can be stored within the memory of the machine.

The original versions of **DOS** were designed to make around 640,000 of these locations available as storage space, i.e. this is where the user programs and data

go. The rest of the address space was set aside for the other items which the PC needed, screen memory, BIOS memory and such like.

Unfortunately it wasn't long before this memory size became a serious limitation, so people started to try to find ways around this.

Expanded Memory - EMS

The first attempt to get around the problem resulted in the design of a system called "expanded memory".

It was noticed that there is a "hole" in the address space of the IBM PC, i.e. a little chunk of it which is not actually used for anything. This 64K (64,000) location area was used as the basis of the expanded memory system. Now, filling that hole with memory means that you only have another 64K, but the clever trick was to add switches to the hardware so that the hole could be used as a window on to a much larger piece of memory.

The idea was that you bought a special card which had a huge amount of memory, perhaps another Mbyte or so, and plugged it into your machine. As well as the extra memory, the card also has switches which can be controlled by the computer to select which part of the larger memory area shows through the hole in the memory space. All a program has to do is select each part in turn, and then read and write it as required.

After a bit of competition between rival manufacturers the LIM (Lotus, Intel, Microsoft) standard for expanded memory became popular. However it did have limitations; If you wanted to copy from one part of the expanded memory to another, you had to keep switching your window about, and running larger programs was still difficult, you could only use LIM for holding data really. An expanded version of LIM, called ELIM was developed, but by now newer processors were becoming available which could directly address larger amounts of memory and so the need for LIM was diminishing.

That being said, you will still occasionally come across programs which need expanded memory, and it is important that you understand how to make it available if the program wishes to use Windows.

One of the advantages of the 80386 and 80486 processors is that their memory mapping features make the simulation of expanded memory very easy to achieve.

LIM 3.2

This was the first popular standard for expanded memory. It was established in 1985 and provides for up to 8 Mbytes of RAM. It splits the 64K hole in memory into four 16K pages, each of which can be controlled independently. This standard, whilst perfectly adequate for the movement of data, is not good enough to permit multi-tasking.

LIM 4.0

This specification arose out of improvements to the LIM 3.2 specification made by Microsoft et al. This standard introduced the concept of "backfilling", whereby you let the expanded memory card take over more than just the 64K hole in the address space. By turning off memory in your PC you could increase the size of the window into the memory, greatly increasing the speed and flexibility.

The LIM memory backfill area can range from the 640K boundary down to the 256K mark. This makes possible multi-tasking using LIM, because the bank switched area is now large enough to hold complete programs. This standard was introduced in 1987 and has become the de facto standard for expanded memory.

Extended Memory - XMS

Expanded and Extended memory sound like the same thing, but are really quite different. Expanded memory uses the trick above to make more storage available. Extended memory is quite different; it is simply memory which is located above the limit which can be addressed by the original processor in the PC.

The original processor, the 8088 or 8086, could address 1 Mbyte. The 80286 could address 16 Mbyte. The 15 Mbytes of memory above the old 1 Mbyte limit was called extended memory, but it came with a price.

When Intel made the 80286 they were very concerned to make sure that all the programs which had already been written for their older processors would still work.

They did this by giving the 80286 a split personality. In what is called **real** or Clark Kent mode the 80286 simply behaves like an old 8086, running exactly the same code but having the same limitations in terms of memory. By performing a certain sequence of instructions you can send the processor into a phone booth to emerge as Superman, able to use extra instructions and address a much larger amount of memory. The snag was that the 80286 took rather too long in the phone booth getting changed, so that flipping from one mode to another took too long to be useful.

One operating system which tried to make use of the additional facilities provided by the 80286 was OS/2, but this system has yet to catch on in a big way, most people preferring to use an 80286 as a very fast version of the original 8086 chip. Windows can make use of the 80286 protected mode operation to provide extended memory but some of its more powerful facilities, particularly with respect to multi-tasking **DOS** sessions, are not possible with this processor.

However, extended memory lives on in the 80386 and 80486 processors which offer vastly improved mode changing and memory mapping mechanisms.

HIMEM.SYS

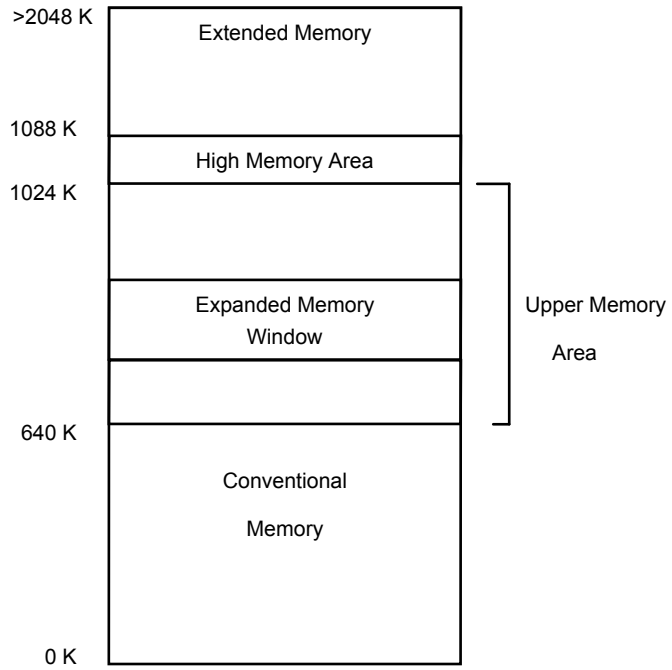
If you want to make use of the extended memory in your machine from a **DOS** program you need something to manage it for you.

The **HIMEM.SYS** device driver provides an interface whereby programs running in the normal "DOS" area below 640K can ask for and be given chunks of higher memory. One of the good reasons for using MS-DOS version 5.0 is that by the use of **HIMEM.SYS** you can greatly increase the memory available for your **DOS** programs. You do this by loading device drivers and some TSR programs into higher memory which is made available by **HIMEM.SYS**.

For Windows operation a **HIMEM.SYS** program is provided to manage the higher memory which Windows needs. The command to load this driver is inserted into your **CONFIG.SYS** file by the **SETUP** program. If you do not load this program, perhaps because **CONFIG.SYS** has become corrupted or reverted to an older version, Windows will not work.

Other DOS Memory Types Explained

When you read about **DOS** and memory you will come across various different letter sequences which refer to types of memory, here is a brief guide to each of them:



Note that in the above diagram the areas are not given to scale.

Conventional Memory - 0 to 640K

This is the ordinary, boring memory which runs from 0 to 640K in the address space. It is the memory which the PC was originally granted when the machine was designed and it is the only memory which "vanilla" **DOS** applications are allowed to use.

Although there is 640K of conventional memory, the maximum memory available for **DOS** programs will invariably be less than this, because **DOS** will use some memory for its own purposes. Each device driver, TSR program and the like steals memory from this area, reducing the amount available for your programs.

DOS 5.0 is of special interest when running **DOS** on 80286 and better machines, because it can put portions of itself in other memory areas and leave more for your programs.

UMA - Upper Memory Area

The Upper Memory Area is that portion between 640K and the 1024K (1 Mbyte) limit of the 8086 family of processors. This part of memory is used in the original PC as the place where the screen memory was reserved, along with space for utility ROMS and the BIOS ROMS.

This address space can be used by 80386 and better processors to increase the memory available to **DOS**.

HMA - Higher Memory Area

HMA is the result of a mistake on the part of Intel. When they designed the 80286 they made it possible for the chip to access the first 64K of memory above 1 Mbyte without running in protected mode. This makes the area from 1024K to 1088k - the first 64K of extended memory - rather special. **DOS** version 5.0 and some programs make use of this extra memory. When configuring a **DOS** application to run under Windows you can control whether or not they are able to use this memory area by setting an option in the Program Information File (PIF) - more on this later on.

Device Drivers and Higher Memory

Some other programs which run as part of Windows and **DOS** 5.0 like to make use of extended memory. They must be run after the **HIMEM** driver has been loaded, otherwise they will fail.

EMM386.SYS

As the name implies, this is a device driver which uses the fancy memory management of the 80386 and later processors to provide an Emulation of Expanded memory.

It supports all the ELIM standard requests for areas of expanded memory. and then uses extended memory to support them. You should make sure that this driver is loaded by your **CONFIG.SYS** file if you want to use older programs which make use of expanded memory (some versions of the Lotus 123 spreadsheet, Word Perfect Word Processor and Autocad package like this form of memory).

If you do not plan to make use of this memory you do not need to load the driver, in fact it is best not to load it because it uses up memory.

SMARTDRV.SYS

Windows version 3.0 was supplied with a device driver which was used to cache hard disk reads. We will go into caching later in the course, it is simply a technique which uses memory to speed up disk drive use. **SMARTDRV.SYS** was loaded immediately after **HIMEM.SYS** and used extended memory as the hard drive cache.

Windows 3.1 uses the program **SMARTDRV.EXE** to provide a much improved caching system (more of this later). The Windows 3.1 **SETUP** program should remove the line from **CONFIG.SYS** which loads the **SMARTDRV** device driver and replace it with a line in **AUTOEXEC.BAT** to load the new program.

Windows 3.1 will work with the older **SMARTDRV**, but some of the fancier facilities will be missing.

Configuring DOS 6

DOS 6 can make use of all the types of memory mentioned above, and you can configure it so that the maximum amount of conventional memory is available for users. This configuration is performed in the **CONFIG.SYS** file. This is not an exhaustive guide to memory management in DOS 6, for greater detail consult the DOS 6 manual.

DOS 6 and Memory

You can find out how DOS is using your upper memory by issuing the command:

mem /c | more

- at the **DOS** prompt **before** you start Windows. Any chunks of memory which are available will be listed at the end of the memory display. You can then use the **loadhigh** command at the **DOS** prompt to load TSR programs into Upper memory and the **devicehigh** command in **CONFIG.SYS** to load device drivers into upper memory.

Note that you can hit situations where improper use of the upper memory can stop your system from running. In this case you should use a "clean boot disk", i.e. a floppy which you have created with a known, bootable, version of DOS 5.0 to allow you to re-load the system and edit the **CONFIG.SYS** and **AUTOEXEC.BAT** files to put things right.

DOS 6 and Upper Memory

DOS 5.0 will make use of Upper Memory and load part of itself into this area, if told to by a line in **CONFIG.SYS**:

dos=umb

This should free some conventional memory for use by **DOS** applications. If you do not load **DOS** into upper memory the HIMEM.SYS extended memory manager (see below) will simply make this area of memory available to any **DOS** application which wants it.

DOS 6 and Extended Memory

Like Windows, DOS 6 is shipped with **HIMEM.SYS** to allow programs to make use of extended memory. This is automatically loaded by the standard **CONFIG.SYS** as installed by DOS 6

DOS 6 and Expanded Memory

Again, like Windows, DOS 6 is shipped with a driver which allows programs to get hold of expanded memory. The driver, **EMM386.SYS,** should be loaded after **HIMEM.SYS** in your **CONFIG.SYS** file.

The MEMMAKER Program

Some PC hardware can provide additional storage space in the upper memory area. This is because if the host processor is placed into "Virtual 8086" mode it can be made to re-map the address space available to programs and use real memory in place of "holes" in the upper memory area. If unused areas of upper memory are locked in this way and then drivers slotted into them, significant amounts of memory below 640K can be freed off.

The first program to perform this function was released by Quarterdeck. A similar program, called MEMMAKER, is shipped with Version 6 of MS-DOS. It probes the upper memory area and then re-configures the driver loading in CONFIG.SYS to map drivers into unused areas of upper memory. It is menu driven, and takes steps to provide a path back if the modification fails.

MS-DOS Virus Infection

A computer virus is a piece of software which is designed to infiltrate computer systems. Sometimes the effects of a virus are just cosmetic, some virus programs are malicious and others cause damage due to defects in their programming.

The user is usually unaware that a virus has loaded itself onto his or her computer system at the time it arrives. It is a characteristic of most viruses that they wait until they have had a chance to infect other systems before making their presence felt. It is possible for viruses to spread via network connections but for the IBM PC the most common method of infection is by running a program which has a virus attached to it, or by booting the system from an infected disk. The PC is much less susceptible to viruses than the Atari ST, Commodore Amiga or Apple Macintosh because it does not automatically run program on a disk which is loaded into the drive.

On the other machines mentioned, simply inserting a disk into the drive causes a program on that disk to be run on the machine. This makes it possible to contract a virus just by taking the directory of a disk. This leads to a need for a program which monitors inserted disks, and ensures that they do not attempt anything naughty.

Just putting a disk into a drive and taking a directory will not move a virus from the disk into your machine. However, if you run a program on that disk it could load a virus into your machine. MS-DOS provides no protection for important system areas. Unlike large operating systems, which limit the number of things a user program can do, MS-DOS lets any program access any part of the machine, including the system disk boot area.

Boot Block Virus

A virus will attach itself to the boot block so that, once installed, it is loaded every time the infected machine boots up. The virus may take steps to hide itself, such as intercepting reads to infected sectors and returning clean ones. It may also load itself into memory once it has booted, and thereby infect the boot sectors of other disks. A *stealth virus* will change its appearance by re-ordering its program code so that search programs will not detect it.

Note that all disks, even ones which are not system disks, have an area which is loaded when the machine tries to boot from them. The message :

Insert system disk in drive A:

- is produced by a tiny program on a data disk, telling the user that this disk cannot be booted from. Boot block viruses will infect this area as well. This means that if you boot from an infected data disk, and see the above message, you have also been infected with the virus.

The general rule to stop yourself being infected with a boot block virus is never boot from a floppy disk. Some PC BIOS chips can be configured to prevent any boot from the floppy disk. If you do not have this option simply make sure that there is no disk in the floppy drive when you turn your machine on.

If you think that you may have a boot block virus your best bet is to run a virus scanner program which will check your boot block against a known good one, or look for the "signature" of known virus programs.

A well known boot block virus is called "Stoned". It does not do anything malicious, just displays a message every eighth time you boot the machine. However, it can become confused when it copies itself onto a floppy and corrupt the File Allocation Table and damage data.

Program Virus

Alternatively a virus will attach copies of itself to other programs on disks which are loaded. These will then copy themselves onto other files on other machines and so on. Some of these viruses are not very clever, and will attach themselves to files which are already infected, causing program files to continually grow in size. A program virus may behave like a boot block virus, and copy itself into the boot block of any disks for good measure.

Such programs like to attach themselves to any program which may be run often, because this gives them regular access to the machine. A popular program to attack is the transient portion of COMMAND.COM. If you think that you may be infected, compare the length of this file with one on a known, good, write protected disk. Some viruses which work like this are known by the length of the extra bit they add on, the 1812 virus is an example of this.

Interrupt Virus

Another way in which viruses load themselves into the machine is via the interrupt vectors which DOS supports. They can then be invoked each time a program calls an interrupt to perform a function, or each time a hardware event

such as a clock tick or a key press takes place. These function in the same way as a Terminate and Stay Resident (TSR) program. They may infect disks which they find in the floppy drive, they may infect each program which is run after they have loaded. There are virus scanners which search memory for the "signature" of known viruses, allowing you to detect when you are infected. Unfortunately the latest trend is towards "stealth" viruses which alter their appearance in memory with each infection, making detection more difficult.

Virus Spotting

Not all problems with computers are caused by virus infection. Before you blame the virus it is important to verify that there is no other cause of your problem. You should be aware that virus infection is possible, but it does not happen particularly often.

The following items are symptoms of virus presence :

- files with strange names appearing in directories
- bad blocks appearing on floppy disks
- write errors on write protected disks which you are reading.
- strange numbers of free blocks on drives, e.g. 730K free on a 360K floppy disk.
- changes in the size of program files.

If you notice any of these there is a possibility that a virus has infected your system. You should keep a copy of a virus scan program on a write protected, bootable floppy disk, boot from this disk and then run the scan program on all disks which you are suspicious of. There is a shareware program called VSCAN from McAfee Associates which is regularly updated with new virus signatures.

Virus Killing

Because viruses can attach themselves to program files, do not think that because you have re-formatted your system disk you are now safe. Ideally you should re-load all your applications from known safe copies (perhaps the original masters).

Do not regard backups which you have taken as virus free, although regular dumps are one way to protect yourself against the damage caused by some of the more vicious viruses.

Ideally use a known virus killer from a reputable company.

Virus Precautions

There are a number of steps which you can take to reduce the danger of virus attack.

- Always boot from your hard disk, do not boot from any floppies which someone has given you.
- Never just run a new program which you have received from a bulletin board or someone else. First scan the program for known viruses. Even then, it is useful to keep a "sacrificial" machine onto which is loaded any newly acquired software. Only after this has suffered no obvious ill effects should the program be turned loose on other machines.
- Write protect all floppy disks if you not really want to write to them, e.g. if loading some data onto a colleagues machine write protect the disk before you load it.

- Try using a slightly different operating system, perhaps DR DOS 5, because virus programs might not be able to make sense of it.

Virus Misconceptions

Finally on viruses, a few things which they are not able to do :

- A virus cannot "live" in a data file. It can damage one, or put "I'm a virus, har har" into the text of your document, but it cannot gain control of the machine from a word processed document file or spreadsheet.
- A virus cannot live in the memory of a machine once the power has been turned off. IBM PC/AT and compatibles have a small chunk of memory which is kept alive by a battery when the machine is not switched on. This data is therefore retained but, because the data is never obeyed as a program, it is not possible for a virus to live there and gain control of the machine at a later time. This memory can be damaged by a virus, but a virus cannot operate from there.
- A virus cannot circumvent the write protection of a disk drive. The write protect is a physical thing which is detected by a sensor in the drive itself. The disk drive cannot write to a protected disk, therefore a virus cannot either. Some people sell "write protect" programs which purport to write protect a hard disk - a thing which is not done on the drive hardware. These may or may not be a good idea, a good virus writer should be able to easily get around a hard disk write protect program.

Windows 3.1

Windows

This is becoming the standard windowed interface for the IBM PC range. Microsoft is also planning to make the windows front end into the standard method of display presentation for other machines.

Window Based Operating Systems

Windowed operating systems grew out of a recognition by various workers in the field that the standard command line interface which had been used previously was not the way that people preferred to work. In particular conventional systems tended to force a user to work on one thing at a time, whereas in reality a user would have a number of things to do at any one time, and may need to switch between them. In addition, there should be no need for a user to memorise large numbers of commands and command formats. Research at the

Xerox Pablo Alto Research Parc led to the development of quite a different user interface based around Windows, Icons, Menus and Pointers (WIMP).

Window

A window is an area of the display given over to one particular process or task, in which a dialogue which concerns that task only takes place. Windows can be moved, re-sized and overlapped, and the windowed operating system will inform the program driving the window that these events have taken place.

ICONS

An icon is a pictorial representation of some program or function which the user can select. The graphic used is designed to represent in a visual way the use to which that program is to be put. In addition icons can be assigned to data files to indicate the type of data which the file contains.

Menus

A menu is a selection of options which are relevant at a particular point in the use of a program. The desired command is selected from the menu, often using a mouse pointer. Each option has a name which allows its function to be identified. For large applications a hierarchy of menus can be built up, with some menus leading to other menus and so on. The design of menu structure is difficult, what seems intuitive to one use may seem the "wrong place" to have that particular option.

Most users however seem to prefer navigating menus looking for a command to searching through a manual for the required function name.

Buttons

A button is an area of the screen which mimics a push button. The user can select this button and "press" it with a mouse button. The button can be used to select various options within a program.

Slider

A slider is a device for moving around an entity or selecting the level of a particular value. By selecting the slider and moving it a message can be sent to the program driving the window telling it that a certain value is changing. It can then take appropriate action.

Mouse Driven

The mouse is used to drive a pointer around the screen. The mouse is fitted with one or more buttons which are pressed to select or activate items on the screen.

Standards and Windows

Windows is becoming a standard on IBM Personal Computer based equipment. In the area of workstations another standard is establishing itself. This is that of X Windows, a definition of a means by which a host machine can control a display and receive input from keyboard and mouse. The crux of X Windows is that it is designed to be independent of the particular hardware in use. Programs which use remote terminals as X Windows devices will run on any compatible

machine. The workstation market place is currently heavily into standards, with standard versions of UNIX beginning to spread across machine types. The X Windows windowing environment may become the standard front end for workstation use.

An Introduction to Windows

MS-DOS is a single user, single tasking operating system with a character based interface. Early versions of Windows attempted to improve on this by adding task switching and a graphical interface.

Windows 3.0 was the first version to achieve large scale popularity. It was designed to run on Intel 286 machines or better, on top of the existing DOS operating system.

Windows 3.1 was a significant upgrade, bringing True Type fonts and improved network support.

Windows for Workgroups positioned Windows as a "one stop solution" for those wishing to install groups of machines which are to share data and resources.

Windows NT is designed to run on high performance platforms. It supports high levels of security and access control and can be run across more than one central processor (for improved performance) and on processors other than the Intel range (for portability). A server version of NT can be used to support networked clients. Windows NT does not run on top of DOS, being an operating system in its own right.

Windows 95 improves on the operating environment provided by Windows 3.1 and also replaces DOS as the underlying system running the computer. The interface to the user is based around *objects*.

Cairo is a future operating system under development by Microsoft which will merge the object environment of Windows 95 with the high performance platform support of Windows NT. The first fruits of this work are appearing, with the release of the Windows 95 user environment for use on Windows NT version 3.51.

In this section we briefly consider what Windows is, and why it is so special. We look at the features which it provides and consider what may happen to it in the future.

Windows is **not** an operating system. An operating system is something which controls a computer at the lowest level, managing resources and making them available to users.

I like to think of Windows as a *Graphical Environment.* It sits on top of an operating system (MS-DOS) and provides the user with an attractive, intuitive interface to drive programs with. To make the environment even better, it also contains a number of useful additional facilities to make best use of the abilities of the newer processors available in many PCs. These include:

- Improved Memory Management. Windows can make available large amounts of memory which programs (applications) can request from it. Windows will manage the memory correctly and ensure that blocks are only accessed by the processes which have been assigned them. This facility is made much easier by the design of the new PC chips, the 80386 and 80486, which have the ability to lock processes into particular memory areas and tell Windows if any violations take place.
- Task Switching and Multi-Tasking. Windows will allow you to switch from one application to another, Task Switching, and also -

if you have an 80386 or better and sufficient memory, to run more than one task simultaneously.

- Virtual Memory. Windows can create the impression of greater amounts of system memory by using the hard disk as a memory substitute; swapping the relevant parts of the program on and off the disk as required. This allows you to run more and larger programs than your computer's main memory would otherwise allow.
- Disk Caching. Windows can make use of computer memory to speed up disk access. It does this by fetching extra data from the disk when it is read, on the basis that other portions of the same area may be needed later. It can also hold data which is to be written to disk, both to allow the writing to take place at a more convenient time and also to allow recently written data to be read quickly.
- Object Linking. Co-operating Windows applications can pass data from one to the other in a seamless way, embedding copies of the data from one application into files for the other and then invoking the required application later as required. "Hot" links can be set up so that, for instance, if the data in a spreadsheet is changed a table in a document which is generated from it can change as well.
- TrueType Fonts. Windows 3.1 gets around the limitations of many printers by supporting a new form of font technology, called TrueType. You can use the TrueType fonts supplied with Windows to get high quality, consistent output on many different kinds of printers, instead of having to rely on the particular printer you want supporting the required typeface. Another advantage of TrueType is that the characters that you see on the screen are generated from the TrueType designs, rather than being pictorial representations of the nearest similar typeface. TrueType fonts can also be embedded in documents that you send to other people, so that you can be sure that they will be able to print your document in exactly the form you sent it.
- Multimedia Support. Windows supports standards for the management of so-called multimedia devices, including sound cards and MIDI keyboards. With the appropriate hardware sounds and live action video can be incorporated into applications.
- Pen Support. Windows also provides a standard for the use of a computer from a pen based interface, rather than the more standard keyboard and mouse approach.

Windows and DOS

Windows will run on top of most kinds of **DOS**, from version 3.1 onwards. However, for best results you should make use of MS-DOS version 5. This has numerous facilities which make it especially suitable, we will look at these later.

The History of Windows

Windows was originally provided as a way in which application writers could produce programs which looked a little like those available on the Apple Macintosh and Lisa machines. It was quite often bundled in with applications which needed a graphical front end, and only rarely sold as a free standing product. It went through two earlier incarnations before being released as Windows 3.0.

This version became extremely popular. Windows 3.1 is a significant upgrade of version 3.0, with a large number of additional features. We will cover Windows 3.1 extensively in this course, with irregular references to version 3.0 when required.

Most new software for the PC is now written to run under Windows, and many computer manufacturers include a copy of the system with their machines. This means that for most users it will the working environment of the future.

Microsoft is working on a version of Windows which will do away with **DOS** completely and take over the running of the entire machine. Windows NT, as this will be called, will also be available for machines other than PC compatibles, moving the Windows environment firmly into the workstation domain.

Windows Program Files

Once you have got Windows mounted on your system you can begin to investigate its inner workings. Windows is made up of a number of component files which are fitted together as the system runs. Sometimes the Windows system will decide for itself which components to use, at other times it will be driven by one of the numerous control files which it uses.

WIN.COM

To start windows you usually type:

```
win
```

When **DOS** sees this it looks through your path for a program with that name and then runs it.

WIN.COM is the program in your Windows directory which is run to start a Windows session. It is simply a loader which checks your system setup and that there are sufficient resources to run Windows. If there are it then sets up the display, puts up the advertising screen (the pretty picture of the Windows logo) and runs the appropriate version of Windows proper. You can add options to your invocation of Windows if you want to run it in particular modes, otherwise **WIN.COM** chooses the most appropriate mode and runs that. The options are :

Standard Mode Option - /S or /2

```
win /2
```

This option will cause Windows to run in standard mode. This mode is the only mode available if you have an 80286 processor in your machine. If you have an 80386 and at least 2 Megabytes of memory Windows will usually try to run in enhanced mode.

However, if you want to get the maximum speed out of your 386 (running in enhanced mode reduces straight line performance) you can use this option to select standard mode when you run Windows. You will not be able to run multiple **DOS** sessions, nor will you be allowed any Virtual Memory.

Enhanced Mode Option - /3

```
win /3
```

This option will cause Windows to run in enhanced mode. This will only work if you have an 80386 or better in your machine, and at least 2 Megabytes of RAM.

Windows Core Files

The Windows core files are each given specific responsibilities within a Windows session. There are three core files, the kernel file, the user file and the gdi file:

Standard Mode:

KRNL286.EXE
USER.EXE
GDI.EXE

Enhanced Mode:

KRNL386.EXE
USER.EXE
GDI.EXE

The Kernel Files

The Kernel files take overall control, managing memory and resources, deciding which program is running and loading and running requested applications. There are two executables, one of which is run by **WIN.COM** :

If you run Windows in standard mode the image that is run by **WIN.COM** is called **DOSX.EXE.**

If you run Windows in enhanced mode the image that is run by **WIN.COM** is called **WIN386.EXE**

Remember that even though these files have got .**EXE** extensions you should **never** try to run them directly. This would not be a clever way of starting Windows quickly, it would be a clever way of stopping your machine quickly!

USER.EXE

The **USER.EXE** kernel file is in change of the interface with the user, translating keyhits, cursor movements and the like into the correct Windows messages, and managing the display of windows on the screen.

GDI.EXE

The GDI (Graphics Display Interface) kernel file is in charge of producing images on the screen. Other Windows functions tell the GDI what to do and it converts these into the relevant commands to drive the particular hardware you have.

Dynamic Link Libraries - DLLs

Windows internal and external functions are provided by Dynamic Link Libraries (DLLs). They work like this:

When a program, or a part of Windows, wants to perform a "Windowy" thing, like draw a character on the screen, move the mouse, fetch an option from a menu etc., it will call a function to do this.

In a **DOS** application this function request would be passed through your application into the part of **DOS** which does that particular job, which then performs it.

In the case of Windows, the portion which does the job is linked dynamically as Windows runs. When Windows builds itself in memory it installs the version of the library which is appropriate to that particular configuration. This means that

the actual code which is performed in response to a function request will change from one installation to the next, but the program which is calling the function will never know this.

Furthermore, you can add new functions to the Windows system by loading other Dynamic Link Libraries which will service these calls. This means that a greater range of functions can be supported, because not all the code to handle them needs to be in memory at the same time.

This means that the actual code which is performed can be easily changed if your configuration changes.

Within Windows a Dynamic Link Library is denoted by the file extension **DLL.**

Note that Windows applications can be supplied as **DLL** files.

Driver Files

A driver file is a bit like a Dynamic Link Library, in that it provides a set of facilities which other programs can use. However, a driver file is not usually dynamic, in that it is loaded once and then used thereafter. Windows uses driver files so that it has a uniform interface between itself and the very wide range of devices which it must use.

The Windows distribution kit contains a large number of driver files which are copied by **SETUP** as required. If you buy an exotic device which is not supported by the standard set up you will need to find a driver for it. Driver files are denoted by their **DRV** extension. Drivers for Postscript printers have the extension **WPD.**

The Windows Resources Kit manual has a copious listing of all the driver files which are supplied.

Font Files

Windows always operates in graphics mode, drawing the required characters on the screen. This means that it must know the designs of the characters it will use on the screen. It also needs to make use of particular character designs when driving printers. We will cover fonts in more detail later, for now you will note that there are a large number of .**TTF**, .**FNT** and .**FON** files in the **SYSTEM** directory in your Windows directory.

MS-DOS Support Files

When Windows is providing a **DOS** environment it makes use of a set of additional driver files to provide the facilities which **DOS** applications will expect, for example a Windows version of the mouse driver is supplied so that **DOS** programs can make use of the mouse when running under Windows.

There are two versions of the part of Windows which manages a **DOS** session, **WINOLDAP.MOD** for use in standard mode and **WINOA386.MOD** for use in enhanced mode.

In addition, Windows makes use of a set of "grabber" drivers which manage the interaction between a **DOS** application and the screen it is using. In standard mode the number of things you can do is limited, because the 80286 processor which this mode is designed for is restricted in how it can manage memory access. In enhanced mode the grabber will allow **DOS** applications to run within windows on the virtual screen and provides comprehensive cut and paste facilities. The grabber files have the language extension .**GBR**, there are standard and enhanced mode versions of these files.

Initialisation Files

As Windows runs, it refers to files which it uses to keep track of the large number of system settings available. The contents of these files control Windows at "ground level", you can make drastic changes to the way that Windows performs just by changing a line in these files.

Understanding how the files are used is crucial to effective use of Windows, but remember that you are being given the keys to a very powerful "Pandora's Box". If you get the settings wrong you can stop Windows from working.

With this in mind you should make changes to these files very carefully, and always keep backup copies of them so that you can easily recover from any mistakes. When you install a new Windows application, it will often make changes to these files as well, so that the Windows system is made aware of it.

This means that before **any** program installation you should copy the settings files, so that if the installation fails for any reason you can very easily put things back to how they were before.

INI File Format

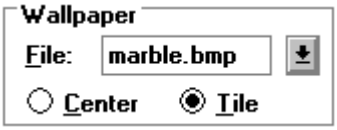
All **INI** files have the same internal format. They are organised in sections, with each section holding details of one aspect of Windows operation. The name of a particular section is given enclosed in square brackets, for example the heading **[desktop]** appears at the start of the section which holds information about the settings for your desk top; what colour scheme is used, what pattern is used for the various areas, etc.

In fact each settable item in the Desktop program you run from the control panel has an entry in this section, i.e. you could get the same effect by changing the settings here as you would by running the Desktop program.

Within a particular section there are a number of settings for that section. Each setting has a name (for example **TileWallpaper**), the equals (**=**)character and then one or more items separated by commas, for example, to return to the **[desktop]** section you will find the item:

TileWallpaper=1

- in this section. This corresponds directly to the :



- option in the Desktop setup.

WIN.INI

This is the file which contains all the major settings information for Windows itself. It specifies how Windows is to behave, and gives details of the hardware configuration in use. It also holds details of the fonts to be used and network connections supported. A full list of the sections in **WIN.INI** is given in the Microsoft Windows Resources Kit manual.

SYSTEM.INI

This file contains lower level information than **WIN.INI** and so should be handled with even more care. The version of **SYSTEM.INI** you end up with is determined by **SETUP**. This is where the various dynamic link libraries and driver files are specified which make up the version of Windows which you are running.

CONTROL.INI

This contains sections which reflect the settings of options you would select from the control panel.

PROGMAN.INI

This file controls how the program manager behaves, how applications are arranged in windows and precisely what happens when they are run. It is also interesting because it contains entries which define which group files are active, and how they are to be displayed.

Group Files - .GRP

A group file is a file with the extension .**GRP** . Windows creates a group file when you make a new group. Some groups are created by **SETUP**. The file itself is binary, i.e. you are not able to change its contents directly, and it is managed by the program manager (so you should never change it at all!).

The group file is an important link between applications which are installed on the system, and the Windows user interface, which wants to give you a simple way of getting hold of it. Windows will look in **PROGMAN.INI** for the location of group files to present to the user. Within the group can be program items from many different parts of your system (for example on different disk drives and in different directories) but they are linked together by this file. To create a group you use the New item from the Program Manager File menu.

INI File Housekeeping

Provided that you follow the above rules about keeping copies of the **INI** files before any major changes, you should have few problems with them. The time you will have most bother is when you partially delete an application and then have **INI** files lying around which Windows may try to make use of.

A Good Trick is to use the powerful **SEARCH** option in Word for Windows to look right through your system and find all the files with the extension **INI**. This is easy to use and has the benefit of showing you each file in turn. You can then use Word for Windows to edit the file, although you must be careful not to save it in anything other than **TEXT** format.

Windows Disk and Memory Management

One of the benefits which Windows brings is much improved memory management. Windows applications can make use of much larger blocks of memory than were available under **DOS**. The Windows memory manager will look after memory ownership and allocation and provides a standard way of getting hold of memory for any Windows application.

If you are running in enhanced mode Windows will use the disk storage to allow more programs to run that would fit into memory at once.

To make the best use of Windows you need to understand how memory is organised inside a PC and how Windows uses the architecture to provide the functions it does.

Windows Operating Modes

How Windows looks when you run it is usually pretty much the same. However the features that you get depend on which of the two operating modes which you are currently using. We have mentioned the **standard** and **enhanced** modes

when we looked at the command which starts Windows running, now is the time to look a little more deeply into what the differences between the two modes are.

Note that the differences, whilst subtle in terms of what the user sees, are very profound in terms of what Windows actually does. You could say that there are two completely separate versions of Windows, standard and enhanced. Note that also, whilst you might think that enhanced is automatically the one to go for, there are often good reasons why standard mode is preferable.

Standard Mode

Standard mode is the only mode available if you are using an 80286 processor in your machine. The limited functionality of this chip means that Windows has to perform differently with regard to several important issues:

- in standard mode only Windows applications can multi-task
- in standard mode the maximum amount of available memory is limited to 16 Mbytes.
- in standard mode **DOS** applications can only run in "full screen" mode.

Multi-Tasking

Before we worry about problems with multi-tasking with Windows in standard mode it is worth getting the terminology sorted out. When you run more than one application on a machine you get two levels of usefulness, task switching and multi-tasking.

Task switching allows you to move rapidly from one application to another, for example you can pop out of your word processor and look up a telephone number in your database and then go back to the word processor again. With a task switching system whilst I am in the database program the word processor is stopped, but this is not necessarily a problem, because I do not want it to run whilst I look up the phone number. Task switching is what most people want, multi-tasking is useful but not vital.

Multi-tasking means that whilst I am in the database program looking up the phone number the word processor is still active, for example I could set the program off doing a long spell check on the entire document and then go off and look for my number whilst the checking took place at the same time.

Any version of Windows will give you task switching and standard mode Windows will allow you to multi-task co-operating Windows applications. When you write a Windows application you arrange things so that your program calls the Windows system at regular intervals to see what is happening. When Windows is called it then has the option to hand control to another application.

You sometimes come across "naughty" applications which "hog" the system and do not hand control back to Windows very often, you can identify these by the fact that everything else stops when they are running. If a naughty application is running, standard mode Windows has no way of stepping in and taking control, which means that multi-tasking relies on the co-operation of the Windows applications.

When you are running a **DOS** application it does not make any calls to the Windows system at all, and so standard mode windows must give it all the processor time available.

Note that this is not a serious limitation if you only want to multi-task Windows applications, and so being stuck with standard mode Windows is not a total disaster, indeed there are occasions when standard mode is preferable to enhanced mode - see later.

Memory

The maximum amount of memory which an 80286 can address is 16 Mbytes. Standard mode is incapable of using any disk storage to increase the amount of memory available, so this represents the upper limit of the memory in standard mode.

Screens and Windows

When a **DOS** application takes control it will often, for the sake of speed, try to talk to the display memory directly. This is fine for a **DOS** program when it is the only one running in the machine, but if you run a **DOS** program under Windows this will be disastrous because Windows will want to drive the display as well. Standard mode Windows gets around this problem by giving the **DOS** program the whole of the screen to play with and effectively stepping into the background whilst a **DOS** program does it stuff.

System Requirements

To run in standard mode your system will need an 80286 processor or better, and at least 1 Mbyte of memory.

Enhanced Mode

In enhanced mode the problems above are removed by the use of additional features which are provided by the 80386 and better processors.

Multi-Tasking

You will start seeing the word virtual a lot now. For our purposes it means "it is not real, but it might as well be".

The 80386 has an additional mode called "virtual 8086" mode. This is when the chip behaves like a number of standard IBM PC processors, each with its own area of memory. In addition the 80386 can switch from running one of these processors to any other, at regular time intervals.

This makes it very easy for Windows to load multiple applications into different virtual machines and then transfer control between them. In enhanced mode therefore, you can have multiple **DOS** and Windows applications and no one program can hog the machine. At regular intervals Windows stops the current virtual machine and runs the next one on the list. This is often called a "time sliced" system, because the amount of time available is chopped into chunks and given to each application in turn.

The down side of this is that the time taken for Windows to come and transfer control is lost to the programs. This can mean a 20% drop in performance, as the system spends a fifth of its time deciding who is going to run next.

If you only want to run proper, well behaved, Windows applications you can get improved performance simply by starting Windows up in standard mode.

Memory

The full 80386 and 80486 chips have 32 address bits, which mean they can address a mind manglingly large amount of memory. This increases the amount of memory which can be used - simply because the processor can handle more.

However, the wonderfulness of these chips does not end there, they can also change what part of memory is used by a particular program, without the program itself knowing. Without going into too much technical detail this means that instead of having to move lumps of memory about they can just swap pointers to the blocks and the program will not know any different. This facility is used in enhanced mode to greatly speed up the Windows memory management.

A further "good trick" is the ability of the 80386 chips to detect an attempt by a program to get hold of a particular memory location. This means that it is easy to put programs into "boxes" and then get control when they try to step outside these areas. This facility makes possible "virtual memory", of which more later. It was provided in a more limited form on the 80286 chip, but works much better on the newer ones.

Screens and Windows

The fancy memory trapping abilities of the 80386 chip (see above) means that Windows can arrange to get control whenever a program tries to write to the screen. When this happens Windows can then map what the program does into a "virtual screen" and then copy this screen into the appropriate area of the display.

What this means is that not only can you have multiple **DOS** applications in enhanced mode but also that they can each run in their own window on the display. However, because of the work which Windows is doing behind the scenes, you may find that such programs display things rather slowly!

Virtual Memory

Virtual memory is a trick which has been played from the very dawn of computing. It comes from a very simple problem, the faster you want the memory, the more you have to pay for it! In a perfect world we would all have several hundred Mbytes of memory plugged into our computers, allowing us to use lots of huge programs. Unfortunately this is just too expensive, for the larger chunks of storage you have to go for a cheaper storage device, in our case a hard disk.

Virtual memory is where we fool our program into thinking that there is a huge amount of memory available, whereas in fact a lot of it is kept on a hard disk. If you think about it, a program is often only using a small amount of the memory of the machine at any given time.

Windows can put the memory which the application is not using onto the hard disk, and load the various parts as required. Each individual chunk of the memory system is called a *page*, and at any given time a number of pages will be held in the memory and the rest will be stored on the hard disk. To manage this we need some help from the processor, which means that you must have an 80386 or better to use virtual memory and Windows must be running in enhanced mode.

The Swapping Decision

A crucial component of a virtual memory system is the bit which decides which old pages to "throw away". Because you only have a limited amount of main memory you must sometimes overwrite older pages with new ones which you now need to use. If the pages are ones you are reading data from (for example the program itself) you can overwrite them with new ones. If the pages you want to overwrite are ones that you have changed (for example data areas) you must ensure that you write them back to disk before you clobber them.

All these decisions must of course be done at great speed, because all the time you are doing this you are not running the users program, so you are slowing things down.

There have been huge tomes written on this aspect of operating system design, the upshot of which is usually that you use the simplest and fastest technique that you can. (and then find that it does not work properly all the time!).

Arrgh! Another computing term. An algorithm is simply a way of solving a particular problem. With computers you usually express your algorithm in terms of a piece of computer program which does the job.

In the case of Windows a Least Recently Used (LRU) algorithm is used, i.e. Windows will swap to disk the pages which have been left alone for the longest time. This algorithm has the advantage of being simple and quick to implement, but if large numbers of applications suddenly want to use lots of pages they all become very "young" and the system will become unstable.

You do not usually see the system crash if it hits problems like this, but you can see things slow down dramatically and the disk drive go berserk. This phenomenon, known as "thrashing", is when Windows is forced to load and discard parts of memory in order to try and keep things going. The only way to keep going in this situation is to shut down as many applications as possible and things should then improve.

The Windows Swap File

The place where Windows swaps the various parts of memory is called the "swap file". It has a particular size, which limits the total amount of memory which is available, and comes in two flavours, permanent and temporary.

The Temporary Swap File

"Vanilla" installations of Windows use a temporary swap file. This means that the file is created when Windows starts, and discarded when Windows finishes. This is all well and good, but this means that Windows will get whatever area of the disk is free when it starts, and this may not be organised very efficiently.

It is also a problem if you start Windows on a machine without much free space. Windows will grab almost all of what is left for the swap file and then you will run into huge problems later on, particularly if you try printing anything!

There are options in the **SYSTEM.INI** file which allow you to control the maximum size of temporary swapfile that you can create and the maximum amount of disk space which Windows must leave available on the disk when creating swap files. They are in the **386Enh** section:

```
[386Enh]
PagingDrive=D
MaxPagingFileSize=3072
```

The Permanent Swap File

The permanent swap file is the best option. This is an area of disk which you reserve specially for use by Windows. In order to minimise the amount of head movement the disk needs to make to get at each portion of the file, Windows will use a "contiguous" area of disk for the swap file, i.e. all parts of the file will be in the same region on the disk surface.

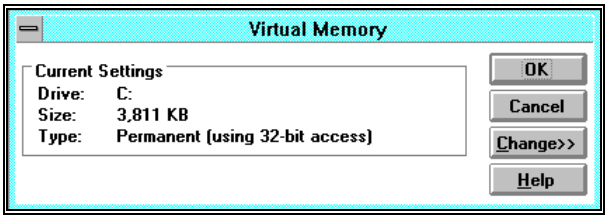
The largest swap file you can have is therefore the largest contiguous area of disk, Windows will tell you this value when you create a permanent swap file.

Creating a permanent swap file is a **must** if you want to get the best out of Windows. It also means that you will not run into so many disk space problems when Windows starts up. A further benefit is that entry and exit to and from Windows is quicker, because the swap file is already there.

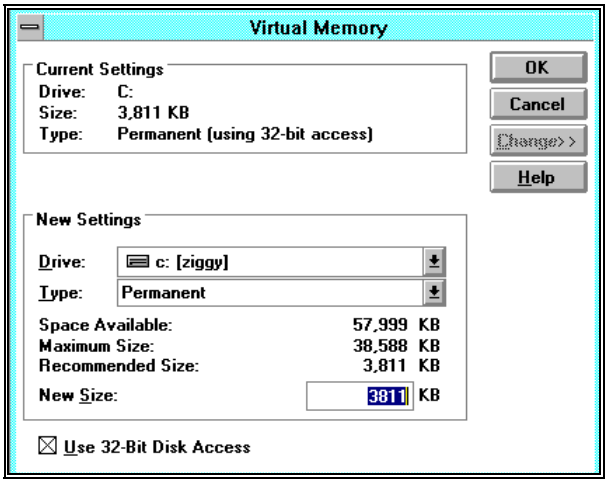
The swap file itself is a hidden file which is held at the root level on the chosen drive - do **not** delete it - particularly when windows is working!

Managing Virtual Memory

You manage Windows virtual memory from the Control Panel. After you select the 386 Enhanced icon you will see that one of the buttons refers to virtual memory, press this to see what the current state of your virtual memory is :



If you press the **Change** button, the window expands to show the following :



This is my normal setup for Windows. To take each portion in turn :

I have a permanent swap file which is 3.811 Mbytes in size. It is held on drive **C**, which is the only drive I have (you can put your swap file on other drives if you like, there can be performance gains if you put applications and your swap files on different physical disks because the amount of head movement required is reduced.

If you click on the selection arrow in the drive box you will be allowed to choose the disk to use for your swap file. If you click on the selection arrow next to the type box the following menu pops down :

Select the option which suits, only a complete maniac would turn off the swapfile by selecting **None**.

If you opt to select a permanent swap file, you should look at the bottom portion of the display which tells you how much disk space you have available and the size of the largest contiguous area. It also gives you a suggested swap file size, which you would be well advised to go with.

If the Maximum Size available is very small, i.e. less than 2 Mbytes, you should run some form of disk "de-fragmenter" to put all the free space on the disk into one single area. Do **not** run such programs from within Windows. Remember to back up your disk first as well!

Once you have selected your virtual memory options you will need to leave Windows and re-start it for them to take effect.

I would also advise you to check the virtual memory box whenever you change the amount of memory in your system, for example if you add 1 Mbyte of RAM to your machine you may find that the optimum size of the swap file will change, and you should let Windows adjust it for you.

Also, if you delete Windows from a machine remember to look for the permanent swap file (called **386PART.PAR**) and delete that as well.

Do not be tempted to set up a swap file in your RAM disk, although this may seem a way of getting a very fast swap system you might as well make the RAM available to Windows directly.

You should also be careful about making a swap file on a network drive. Although this is possible it can lead to very heavy network traffic and should be avoided.

There is also an option in the box to turn on and off 32 bit disk access, we will deal with this later.

Disk Caching

As mentioned above, disk caching is a trick which speeds up the use of your hard disk. Essentially, the disk cache system uses memory to hold data on its way to and from the disk drive. Most of the overhead involved in a hard disk access is the time it takes to move the head to the required track and then the time it takes for the disk to spin to the correct position. The actual time it takes to read or write the data is very small, and you can read or write twice as much data with only a very small increase in the time taken.

Moreover, since disk reading and writing takes place in parallel with the program you are running, the trick is to try to change the use of disk into fewer read and write requests which are of larger chunks of data. The rest of the time you try to keep the program happy by using data which you have stored in your cache.

There are two forms of disk cache, a read cache and a write cache.

Read Caching

The disk caching software in Windows 3.0 only supported read caching. Whenever a program asks for some data from a disk the cache program fetches more than is required, on the basis that the next part of the file may well be the next bit we want. This means that when the program asks for the next part, rather than having to ask the disk drive for the data, the cache software just supplies the data from memory, which is much quicker.

You can get very good performance increases with a comparatively small cache, as the amount of disk head movement is reduced dramatically (you can actually hear the improvement!).

Read caching is good because it is inherently safe, there is no chance of you losing data because we are only affecting data reads.

Write Caching

With Windows 3.1 came a new version of the disk caching program which also supports write caching. Write caching is a trick where you buffer up the write requests and then perform them all at once. This is more efficient, because the disk cache software can write a very large amount of data, and there are occasions when you may want to read data which you have just written (for

example if you were implementing Virtual Memory you would be writing and reading swapfile pages). If the data you have just written is in the cache, you can get it back very quickly.

Also, if you make a series of changes to a part of the disk you simply change the contents of the write cache and can therefore eliminate complete write actions.

The bad news is that write caches can lose your data. If the machine crashes before the write has actually taken place you will lose that write. This is especially dangerous, because the program which is writing the data thinks that the write has worked, and so has no way of knowing that the data has not got there.

Fortunately Windows 3.1 is quite stable, and will not tend to crash without warning. However, I have hit problems when developing my software under Windows. I use a technique called "auto save", where the language system saves the program to disk before running it. This is very sensible, and until Windows worked well. However, if I use write caching I can find that my program can crash Windows before the cache has been written to disk! This is very annoying...

You can also hit problems if you turn the computer off as soon as you have left Windows and got the **DOS** prompt back - perhaps the cache has not been written back. One way of getting around this problem is to tell the disk cache program to empty all buffers and bring the disk drive up to date. You can do this under Windows 3.1 with the command :

smartdrv /c

Alternatively you could wait five seconds, because this is the longest that the cache will allow data to stay in memory before updating.

Another way of getting around this problem is to call Windows from a batch file which performs the above command before quitting.

Note that this problem with write caching is one of the reasons why Windows takes control of the normal reset sequence. If you use this you are given the option to abort the application which is currently running. This gives everything time to settle down before the reset actually takes place. Note also that this means you should **never** attempt to reset the computer using any reset switches and the like.

SMARTDRIVE

This is the program which is supplied with Windows 3.1 to look after disk caching. A version of **SMARTDRIVE** is also shipped with DOS 5.0, although this is the earlier, device driver, version. DOS 6.0 has yet another version of **SMARTDRIVE** which can be executed as a program. You should use the newest version of the program that you have.

When Windows installs it can alter your **AUTOEXEC.BAT** to load **SMARTDRIVE** automatically. You can find out all the options to **SMARTDRIVE** by issuing the command:

smartdrv /?

- at the **DOS** prompt. You should see the following :

```
Installs and configures the SMARTDRIVE disk-caching utility.
smartdrv [[/E:elementsize] [/B:buffersize] [drive [+]|[-]] [size] [winsize]]...

drive letter    Specifies the letter of the disk drive to cache.
          (drive letter alone specifies read caching only)
+          Enables write-behind caching for the specified drive.
-          Disables all caching for the specified drive.
size          Specifies the amount of XMS memory (KB) used by the cache.
```

```
winsize       Specifies the amount of XMS memory (KB) used in Windows.
/E:element size Specifies the size of the cache elements (in bytes).
/B:buffer size  Specifies the size of the read buffer.
/C            Writes all write-behind information to the hard disk.
/R            Clears the contents of existing cache and restarts SMARTDrive.
/L            Loads SMARTDrive into low memory.
/Q            Prevents the display of SMARTDrive information on your screen.
/S            Displays additional information about the status of SMARTDrive.
```

You can use some of these commands to change the way that **SMARTDRIVE** behaves. You can only set the amount of memory which is used for the cache the first time that **SMARTDRIVE** is loaded.

By default Windows will set **SMARTDRIVE** to use all the extended memory in the machine. This is fine for Windows, which tells **SMARTDRIVE** to give some memory up when Windows starts, but is a problem if you want to use programs in **DOS** which want to use some extended memory; before they can get any they find that **SMARTDRIVE** has got the lot! The only way around this is to reduce the amount which **SMARTDRIVE** starts with by changing the value when you first load it. Remember that if you do get some more memory you must change the **SMARTDRIVE** load instruction if you want to make use of it for caching.

The default Windows installation only performs write caching on the hard disk drives, only read caching floppy ones. You can force **SMARTDRIVE** to cache writes to floppies if you like, but this is not advisable because you might remove a floppy disk before it had been written to properly. Also, the kind of use to which floppy drives are put does not usually require caching.

Using 32 Bit Disk Access

When Windows is running Windows applications it will use the most powerful mode of the processor in your machine, this is usually called "protected mode". However, whenever Windows wants to talk to the disk drive it must use **DOS** to do this, and must therefore drop back into the slower mode of the processor to run the **DOS** code which talks to the disk drive.

One of the advantages of protected mode is that it can transfer things in 32 bit chunks, which allows for faster data movement between memory and processor. One way of speeding up disk access would be for Windows to talk directly to the disk hardware, bypassing **DOS** completely and thereby transferring data much more quickly.

The good news is that this can considerably improve disk speed, the bad news is that the disk hardware must be highly compatible with Windows, any strange incompatibility will cause not only a system crash but can also damage all the data on your hard disk.

When **SETUP** installs Windows it checks to see if the hard disk on the machine might be compatible with 32 bit access. If it thinks that it should work it allows a "Use 32 bit access" check box to appear on the virtual memory management screen. If you set this box, next time Windows starts it will try to use 32 bit disk access.

This is one occasion when I managed to blow up Windows completely on a slightly incompatible machine, and the problem is that you cannot get Windows to run again to clear the check box. At this point you must resort to editing the **SYSTEM.INI** changing the following line in the **386Enh** section.

```
[386Enh]
32BitDiskAccess=on
```

to

```
[386Enh]
32BitDiskAcess=off
```

This should allow you to get the system running again.

Controlling DOS Applications

Windows is a good place to run **DOS** programs, particularly if you want to run more than one at a time and easily switch between them. The hard part is setting up an environment in Windows within which the **DOS** application will happily do its stuff. Windows provides very flexible mechanisms for customising what the **DOS** program actually sees, and with persistence and ingenuity you can get most **DOS** applications to run successfully in a Windows environment.

Windows DOS Environments

When a Windows system runs a **DOS** program it tries to step aside and give the **DOS** application the impression that it alone is running in a free-standing PC. How effectively Windows can do this depends on the processor, as we have seen above.

The Program Information File - PIF

When you run a **DOS** application Windows needs to know how it is to be treated - how much memory of what kind to give it etc. It does this by looking in the Program Information File for that application, or using the default one.

A set of ready made **PIF** configurations is included with Windows in the file **APPS.INF** file. This is the file which **SETUP** uses when scanning your hard disk for matching programs. Most modern **DOS** applications are shipped with a **PIF** for windows use, which you can just install.

Unlike the .**INI** files a **PIF** is a binary file, so you cannot change it using a standard editor. Instead a program called the **PIF** editor is supplied as part of the Windows system. You run the **PIF** editor from the control panel. What you get depends on whether you are running in standard mode or enhanced mode:

Windows and Printing

Windows provides a very flexible handing of printers and print output. You can have several different printers attached to your machine, either directly or via the network and Windows makes it very easy to switch between them. If you chose the correct font sets you can also be sure that the output which you produce will be of the highest quality possible on the particular device you have chosen.

The Print Manager

The Print Manager is the program which is charged with looking after the print operation. It is started whenever you begin a print job under Windows. Once it is running, you can ask it about the progress of a given print job, and manipulate entries in the queue of items to be printed.

You do not have to use the print manager if you don't want to, the printers icon in the Control Panel activates a menu which allows you to turn the stop Windows from using the print manager when printing.

Turning off the print manage is a good idea, particularly if you have a slow machine with limited memory. It is often difficult to work whilst printing,

because the machine itself runs so slowly, and spooling to a hard disk can generate a lot of hard disk activity.

Spool Queue

The spool queue is managed by the Print Manager. It is the spool queue which makes background printing possible. When an application prints to the Print Manager, the output to be sent to the printer is saved on disk in the spool queue area. The Print Manager then sends the data out to the printer, at the rate which the printer can accept it. This means that you are not left waiting for a slow printer, you can resume using your application as soon as it has finished sending its output to the spool queue.

This spooling is performed at the expense of disk space. If your print job contains lots of pictures, which take up large amounts of space, you may find that Windows runs out of disk space whilst printing.

In this case you should turn off the Print Manager and then try again. Remember though, that Windows can use up a lot of disk space and memory just assembling the picture to be printed, and so you may still run into problems.

Drag'n'Drop Printing

You can use the File Managers Drag'n'Drop feature to print a file without explicitly loading the application which is to print it. To do this Windows must be aware of the correspondence between the type of the file you are picking up and the application which uses that form of file. This mapping is set up in the **WINDOWS.INI** file in the following way :

```
[Extensions]
doc=C:\WINWORD\winword.exe ^.doc
dot=C:\WINWORD\winword.exe ^.dot
rtf=C:\WINWORD\winword.exe ^.rtf
cdr=c:\coreldrw\coreldrw.exe ^.cdr
```

The extensions section links file extensions (the three characters given after the "." in a **DOS** filename) with particular applications. In the example above, the **.DOC**, **.DOT** and **.RTF** extensions are linked with the Word for Windows (held in the program file **C:\WINWORD\winword.exe**) and the **.CDR** extension is linked to Corel Draw. Note that this illustrates that when an application installs itself it will add to the entries in the extensions section which link it to files which are of the required type.

If, in the File Manager, you double click on a file Windows uses the extension to decide what program to run to work on that file. In the same way, if you drag a file from the File Manager to the print icon, the extension is used to decide which program to run to print the file.

Printing and Fonts

Windows supports a very wide range of printers, and can drive them to produce output in many different styles and sizes. Windows 3.1 improved on earlier versions by bringing the TrueType font sets to augment the existing build in Windows fonts and printer fonts.

The words font, typeface, and character set seem to have different meanings depending on who you talk to. Before we go any further we should consider what the words are to mean for the rest of this document (and for most Windows documentation).

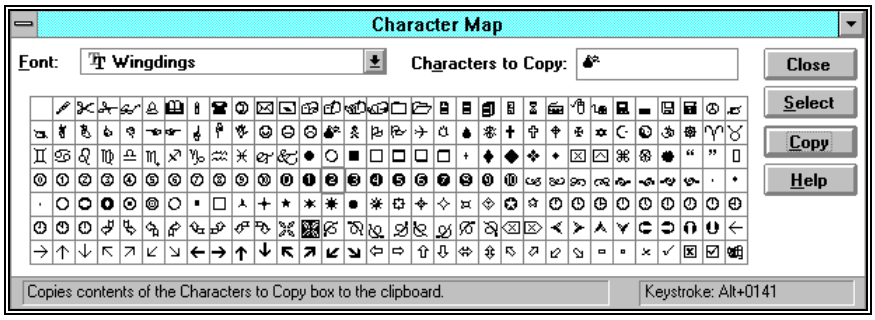
Typefaces and Fonts

I would say that a typeface is a particular *design* of characters. This design runs across the various styles in which the characters can be produced, for example you can have bold and italic characters in a typeface which are derived from the original design. The typeface will have been designed and given a name, and is frequently copyrighted by the designer. Examples of typeface names are Times Roman and Helvetica. Note that with a "proper" font the bold and italic versions will not be enlarged or slanted versions of the same base design, but are completely different designs.

You can group together typefaces into particular font families which share certain characteristics, for example typefaces which have not got any Serifs (twiddly bits on the ends of the straight parts) are grouped together. Windows recognises the existence of five font families, Roman, Swiss, Modern, Script and Decorative. It is important that you remember this, because Windows will sometimes look for a font from a matching family if it finds that it cannot print with the particular font that you have selected.

A character set is a particular mapping of keystrokes and the numbers they represent onto a particular shape of character. The character set you will have heard most of is the **ASCII** (American Standard Code for Information Interchange) and this is the most popular one for mapping character codes onto particular marks on the paper - for example the **ASCII** code of **65** (the one you get by holding down **shift** and pressing **A** on the keyboard) maps, not surprisingly, onto the character **A**. This is all well and good, and for straight text **ASCII** works very well. However, when you get to exotic characters like pound sign and certain punctuation signs, you can hit problems. The **DOS** installation can load different translation tables depending on the keyboard you are using and the display you would like, and from within Windows you can re-map the characters as well.

You can find out how to get particular characters into Windows applications by running the Character Map accessory. This puts up a map of an entire Windows character set for that font and then allows you to select characters from it and place them in the clipboard to be pasted into other applications. You are also told the key sequences you must use to get that character in that particular font. In the example below I have selected the font map for the TrueType font called **Wingdingz**. You can use this to incorporate all kinds of exotic characters in your text. By selecting the bottom right hand character I can incorporate the familiar character into my text.



In the example screen above I have already selected one character which will be copied into the clipboard and am about to select another. The keystroke given in the bottom right box means that I must hold down **ALT** and then press the numbers **0141**. Note that for this to work you must use the number keys on the numeric keypad on your keyboard, and the NUM Lock light must be lit. You can prove that this works by holding down **ALT** and pressing **65**, you should be rewarded with a capital **A**!

Raster vs. Vector Fonts

The early computer fonts were designed for a particular display type, or printer mechanism, in which the characters were made up of dots of a particular size. The characters were expressed as a map of bits which were to be set to represent that particular shape. If you wanted a different size of the same character design you would have to create a new bitmap. Furthermore you cannot rotate bitmap fonts, only print them out in the orientation they were designed for. Windows can use bitmap fonts to drive printers which support them, for example the Hewlett Packard Laser Jet II series.

The vector font allows fonts to be any size and orientation by expressing the shape as a series of equations which mathematically define the path of the outline of it. You can very easily scale and rotate the shape by applying the appropriate mathematical transformation. You only convert the outline designs into patterns of bits just before you drive the printer or the display. To do this you need a powerful processor to perform the mathematical work to generate the map of bits which actually get printed. Windows is shipped with three vector fonts, Roman, Modern and Script. These can be useful for driving output devices such as plotters and for scaling output on raster devices, but they have been rendered largely redundant by the TrueType fonts.

One problem with vector fonts is that when you scale them you can get corruption of the font design as it is mapped onto low resolution output devices. This happens as the characters on the screen get smaller, lines can appear too thick or disappear altogether. To get around this you must add "hinting" information to the font to tell the scaling algorithm what to do as you approach the limits of the display.

Screen and Printer Fonts

When you are using Windows there are two places where the font designs are being used - on the screen and on the particular printer. Windows separates these two jobs, this allows you to manipulate characters in fonts for which Windows has no display version, what happens is that Windows will find out the font family of the printer font and then use a display version from the same family. Some fonts are supplied for both the screen and the printer.

Windows uses a particular set of fonts for the "system fonts", i.e. the ones which are used for all the Windows user interaction. Note than in version 3.1 you are able to change this font for a different one, although this is highly inadvisable!

Postscript Fonts

Postscript was the first popular standard for expressing fonts in terms of vectors. It was designed for use with printers which support the Postscript page description language. The Postscript page description language is a programming language expressly designed for applications to drive printers and the font standard is part of the language specification. You can send a Postscript font into a Postscript printer, or you can use one of the ones which are built into the printer.

If you use Windows you will sooner or later come across a Postscript laser printer which you will need to talk to. This form of printer is the most powerful and flexible on the market, and the Postscript language is also used to drive very high quality typesetting machines.

A Windows enhancement called Adobe Typeface Manager allows you to use Postscript fonts for your screen display.

TrueType Fonts

TrueType is an invention of Microsoft and Apple. It is designed to allow you to send output to any printer and be sure of getting characters the right shape and style. A TrueType font is a vector font with hinting which can be used for both display and for driving a printer. Whatever the printer, Windows can use a TrueType font to get consistent output. You can also bundle a TrueType font in with a document so that anyone who receives that document can print it out in the font you created it.

If you want to be absolutely sure of being able to print your output on any printer you should use a TrueType font to prepare it. Windows is supplied with a few TrueType font families, you can buy and install other ones as required.

You can even restrict the list of available fonts which Windows will display to the TrueType ones supported on your machine, to guarantee uniform output.

TrueType is used for both the screen output and for the printer display, Windows will scale the fonts it uses for display purposes to match the choices you have made in your application. If you decide to go for large numbers of sizes and types of font, you will find that Windows slows down and starts using up disk space and memory for the font layouts. You should therefore restrict yourself to a few sizes and styles of font for your documents (this will make them look nicer too!)

If your printer supports dowloadable fonts, (i.e. if you can load raster designs into the printer and use these to print text) Windows will convert the TrueType rendition of the character into a downloaded version and send the designs into the printer. If the printer does not support these Windows will render a complete bitmap in the memory of the host PC and then send this into the printer.

If the printer is a Postscript printer Windows will translate the TrueType expression of the font into a Postscript version and then send this to the printer. It will then use these fonts to print the text.

Windows 95

Evolution of Windows 95

MS-DOS

MS-DOS is a single user, single tasking operating system with a character based interface. Early versions of Windows attempted to improve on this by adding task switching and a graphical interface.

Windows 3.0, 3.1 and Windows for Workgroups

Windows 3.0 was the first version to achieve large scale popularity. It was designed to run on Intel 286 machines or better, on top of the existing DOS operating system.

Windows 3.1 was a significant upgrade, bringing True Type fonts and improved network support.

Windows for Workgroups positioned Windows as a "one stop solution" for those wishing to install groups of machines which are to share data and resources.

Windows NT

Windows NT is designed to run on high performance platforms. It supports high levels of security and access control and can be run across more than one central processor (for improved performance) and on processors other than the Intel range (for portability). A server version of NT can be used to support networked clients. Windows NT does not run on top of DOS, being an operating system in its own right.

Windows 95

Windows 95 improves on the operating environment provided by Windows 3.1 and also replaces DOS as the underlying system running the computer. The interface to the user is based around *objects*.

Cairo

Cairo is a future operating system under development by Microsoft which will merge the object environment of Windows 95 with the high performance platform support of Windows NT. The first fruits of this work are appearing, with the release of the Windows 95 user environment for use on Windows NT version 3.51.

Windows 95 Features

Windows 95 and MS-DOS

Windows 95 does not run on top of DOS any more. DOS is supported by an environment which runs within Windows 95 and "pretends" to be a DOS box. The low level operation of the system is now controlled directly by Windows 95, which controls access by other applications. Some DOS applications which run by acting directly on the hardware may not execute under Windows 95, but such applications are becoming the minority.

Windows 95 Tasking

In Windows 3.1 applications co-operate over who has the processor, i.e. a given application will return execution to the windows system when it has finished a time slice. This allows badly behaved applications to stop others from running.

Windows 95 will not allow this to happen, programs are scheduled "pre-emptively". i.e. the system does not wait for an application to hand back execution, it interrupts it.

Windows 95 Application Program Interface

An *application program interface* is the way in which programmers request information and services from an operating environment. Windows 3.1 provides an API based around the 16 bit architecture used on older Intel microprocessors. Windows 95 allows the use of both this and the more powerful 32 bit API supported by Windows NT. This raises the possibility of applications which run unchanged on Windows NT systems and Windows 4.0 systems.

Applications running using Win 32 API calls also have individual input streams. Under Windows 3.1, all input is transferred through a single input stream. If a single application stops removing input all others are unable to receive anything. With multiple input streams this problem does not occur, applications can continue running if one of them stops accepting input.

Windows 95 Networking

Unlike Windows running on top of MS-DOS, Windows 95 makes the network part of the operating system. These means that all aspects of network connection are now managed from within Windows.

Windows 95 has been positioned as a desktop network client, although it can also serve resources to other machines. However, it is difficult to enforce any security or permissions on resources shared by Windows 95 systems.

Security can be added by making use of a security server, either a Windows NT system or a Novell Netware server, which can validate access to the system and server secure resources.

Windows 95 “Plug and Play”

Plug and Play is an industry wide attempt to make hardware management easier. The operating system will eventually be able to recognise particular hardware and supplied drivers will integrate smoothly with no user intervention.

Plug and Play also provides support for plug in devices which may be added to a system when it is powered up, for example PCMCIA devices which are popular on notebook computers may be inserted and removed without first powering down. For example, Windows 95 will detect the removal of a network interface card and the insertion of a modem card. It will also support live connection of notebooks and docking stations.

For this to work hardware vendors must set standards for plug and play equipment and then adhere to them. The low level software on motherboards and at the bottom of operating systems must also be able to detect particular items and arbitrate if two require the same resource.

This will require modification to the low level BIOS on all motherboards, along with changes to the design of expansion cards.

Older devices, (i.e. non Plug and Play), must also be recognised by the software which then must react appropriately.

Windows 95 and Mobile Users

Dial Up networking is designed to allow you to contact other systems and make use of their data and resources. These facilities are currently provided by local area networks but are restricted to the area over which network connections are available.

Dial Up Networking seeks to provide similar functions over wide area networks, via public information carriers. This will allow you to make use of data and

software via dial in lines, although the limited bandwidth may restrict you transferring data; running applications which load parts of themselves as they execute will be very slow. It is possible to automate the dial in process, so that activating the resource available in this way causes the network connection to be made for you.

The *briefcase* is introduced as a mechanism by which file systems on different machines can be synchronised when the two systems are linked together. For example you may wish to update your hard disk with the files which have changed on the office network, whilst simultaneously adding your changes. By use of timestamps Windows 95 is able to ensure that only the most recent copies of data are used.

Windows 95 User Interface

Objects

Windows 95 can be described as an "Object Based" operating system.

All items within the operating system are connected to the user in the same way as "objects". The way in which "objects" are managed, manipulated and stored is common to all objects, although the precise function of each object may be different.

Objects can be grouped together into folders in the same way that files can be grouped into directories, but with Windows 95, different objects can be stored in the same folder.

A particular object within Windows 95 will have a particular *type*. Each type will have at least one application linked to it. The user is unaware of the application behind the object, simply selecting the object brings the required application into operation automatically.

Object Properties

In addition, all objects have particular properties which can be managed in a uniform way. The precise properties of the object will vary from one object to another.

For example the properties of a printer are different from the properties of a file, but the two objects are identified by a particular name and icon, and the properties are accessed in just the same way

This contrasts with the approach used by Windows 3.1, where objects are managed in different parts of the system, using different conventions. Consider the different way in which you will manage the:

files on the system

printers on the system

system desktop

To manage each of these items in you must load the relevant management program. In Windows 95 the properties of each of these objects are accessed in a completely standard way.

New objects which are added to the system must conform to the object model, and provide appropriate "hooks" so that Windows 95 can activate their property management routines.

Object Storage

Objects are placed into folders which resemble DOS directories.

However, the writers of Windows 95 have recognised that many users find hierarchically structured directories difficult to manage. The concept of placing one folder inside another folder is something which people find difficult to come to terms with.

For this reason, Windows 95 offers the idea of shortcuts, where a direct path to an object in a directory structure can be specified. This way, particular objects in use at any given time can be placed directly on the desktop and accessed by means of a shortcut.

Windows 95 manages a shortcut in a similar manner to that by which Windows 3.1 produces an icon for a particular application program, although in the case of Windows 95 the shortcut itself is stored as a separate file.

The idea of placing icons directly on the desktop is also new to Windows 95. Windows 3.1 program icons must reside in program groups looked after by Program Manager. Files in Windows 3.1 are accessed by File Manager. In Windows 95 this distinction is removed. Files and applications live in folders on local or distant machines.

Selected files and applications (the ones you need to access most frequently) can be placed directly on the desktop and activated from there. Rather than move the original, a pointer to it can be placed on the desktop instead, which means that when you no longer need rapid access to something you can just delete the shortcut. Each user can have their own desktop arrangement on a particular machine or the desktop arrangement can be stored on a central server and loaded over the network when the user logs in.

Windows 95 Users

Each Windows 95 user has their own particular identity.

For a user to gain access to a system running Windows 95, they must first specify their user name and a password. This allows Windows 95 to maintain an identity for each user of the system.

The identity extends to desktop preferences, network connections, the mappings of particular drives to network servers and printer settings. In addition, if a user logs in making use of a password, this password can then be propagated by Windows 95 onto any network servers to which that user requires access. This makes it possible for users to specify a single password at the beginning of a session and make use of other systems on the basis of that password.

The password is stored on the Windows 95 system in an encrypted file, so it is not possible for other users to make use of this password. Note that although Windows 95 makes use of username and password validation to arbitrate entry to the system, it is unlike Windows NT in that the file storage system in use is not able to support any kind of file or directory security.

Once a user has gained access to a particular Windows 95 system it is not possible to restrict their access to the files on that system. However if that user makes use of network resources supported by Windows NT or Novell server systems, the security mechanisms available on these server would be based around the user name and password given by the Windows 95 user.

The password issued by a user when starting a Windows 95 system can be used to validate access to network resources. It can also be used to access a “password file” which allows username-password pairs to be allocated to particular servers and used as required. In this way the user only has to issue a single password to gain access to all their network resources.

The Windows 95 Workplace

In order to make Windows 95 more intuitive several key applications from Windows 3.1 have been replaced by a single uniform interface.

File manager and program manager have been removed and replaced by an environment called “your computer”. From within this the system can be set up and files and network connections attached to the computer can be browsed directly.

In addition an area called “your network neighbourhood” allows often used network connections to be grouped together separately from the entire network to which a work station may be connected. Applications can be activated directly from objects they have created, in the same way as in Windows 3.1 File Manager.

Selecting files is a mechanism for starting the application behind them. Links can be set up between file types and applications so that the appropriate application runs when the file is selected. New applications can add information about themselves when they install, so that their file types can activate them appropriately.

The Task Bar

In Windows 3.1 you can reduce an application to an icon, and thereby free up screen space for active applications. The problem with this is that users tend to lose icons behind active windows. This leads to further copies of the application being loaded, because the original cannot be found.

Windows 95 replaces this minimisation process with a device called the Task Bar.

The Task Bar provides a common point onto which applications are placed when not active. There are several well-defined mechanisms for recovering the Task Bar and reactivating applications placed on it. The Task Bar also provides a mechanism for starting applications, for example if no object produced by that application exists to be activated.

Windows 95 Architecture

Windows 95 and MS-DOS

Windows 3.1 uses Microsoft MS DOS as a platform upon which it runs. All file access and many application program facilities are provided by the underlying MS DOS operating system.

Windows 95 does not work in this way. The low level MS DOS is replaced by a 32 bit operating system. It provides an emulation of the MS DOS environment for those applications which require it, and also has an MS DOS fall back mode in which only MS DOS components run for applications that require a very close conformity to the MS DOS environment..

When Windows 95 boots it takes control as soon as the system starts running. There is no underlying MS DOS beneath the system.

Windows 95 and Applications

There are essentially three kinds of applications which Windows 95 has been designed to run. The way it runs each type is slightly different.

MS-DOS Applications

Windows 95 must run as many MS-DOS applications as possible. These applications must be "unaware" that they are running within the Windows environment. To such programs the underlying system must be as close to a traditional PC running MS-DOS as possible. To add to the complication, because MS-DOS applications are used to having direct access to the hardware, Windows 95 must allow such access to take place and have the desired effect. Furthermore, Windows 95 must prevent such access as might cause damage to the system, for example an attempt to write directly to the hard disk device.

WIN 16 Applications (Windows 3.n)

Windows 95 must also run older Windows applications written for the WIN 16 Application Programmer Interface (WIN 16 API). Such applications are used to running on a machine which has Windows 3.1 or similar sitting on top of the MS-DOS applications. They must also be presented with a suitable environment which does not differ from the original.

WIN 32 Applications (Windows 95 and Windows NT)

In the future most applications will be written for the WIN 32 Application Programmer Interface (WIN 32 API). This is a new, 32 bit wide, interface by which applications and the Windows system interact. It provides many additional facilities, such as multiple threads of execution, demand paged Virtual Memory and linear address space. The WIN 32 API supported by Windows 95 is a subset of that available from Windows NT, leading to the possibility that a single application can run unchanged within both operating systems.

Windows 95 and the Intel Processor

"Real Mode" Operation

The original 8086 processor operated as a machine with 16 bit registers and a 20 bit address space. All further versions of the Intel processors based on this chip start operating in this mode. This mode is commonly referred as REAL mode.

Older drivers and hardware need to operate when the processor is running in this REAL mode. Software written to run in REAL mode cannot operate in any other mode. MS DOS applications normally only ever run in REAL mode.

Very badly behaved MS-DOS applications, such as games, can be run by shutting down Windows 95 and re-starting the system in MS-DOS mode. When you have finished running the naughty program you must then re-load Windows 95 from scratch.

Processors after the 80186 had additional operating modes. The 8286 had limited protected mode, but was never popular. The major problems with OS/2 sprang from a need for the operating system to function on the 80286.

Although the aim of Microsoft is to make all applications operate using the Win 32 application programmer interface and use the processor in its 16 bit mode of operation, there are still some central components in Windows 95 which operate in 16 bit "real" mode. There are two main reasons for this. The first is, this way the operating system can retain compatibility with those applications which require 16 bit operation. The second reason is that code written to run in 16 bit mode is significantly smaller than 32 bit code. This means that Windows 95 will be able to run on systems with smaller amounts of RAM. It is the stated objective of Microsoft that Windows 95 will run comfortably on a machine with only 4 megabytes of memory and keeping the size of the system Kernel down will make this more likely.

All the time that Windows 95 is running the processor is constantly changing from 16 bit to 32 bit operation and back again. This transition is known as Thunking. Microsoft has taken considerable amounts of time and trouble to ensure that the process of thunking takes as few processor instructions as possible. Thunking is also performed by systems like Windows NT when they run 16 bit applications within their 32 bit system.

"Protected Mode" Operation

The 80386 provides a very powerful protected mode which also allows the processor access to 32 bit registers and an extended range of machine code instructions. Windows 95 has been written to run in protected mode and make use of these extra processing facilities to improve the reliability and performance of the system.

During the boot process Windows 95 switches the processor from real mode to protected mode so that it can perform most of its operations in a 32 bit mode.

However, if required to run code written for MS DOS it is also able to switch back into REAL mode.

Note that Windows 95 provides a similar 32 bit environment to Windows NT, but security is **not** rigidly enforced between applications in Windows 95.

Within the 80386 four levels of protection are available. Each of these provides a different level of access to the underlying hardware and other system features. Each of these levels is referred to as a "protection ring".

Windows 95 only makes use of the 2 extreme levels ring 0 and ring 3.

- Ring 0 is where the core system of Windows 95 runs.
- Ring 3 is where the applications run.

The ring protection level at which a program runs has determines the access it is allowed to the hardware and the range of instructions available to it.

Virtual Memory in Windows 95

Within Windows 95, 32 bit applications see memory as a single large segment, using the full 32 bits of the processor to address 4 Gigabytes of memory. There are no segments or offsets as when the processor runs in real mode.

Windows 95 makes use of the ability of the processor to re-map memory and perform a translation between the address which the program is issued and the actual hardware address used. "Virtual Memory" is the range of memory which the program thinks is present.

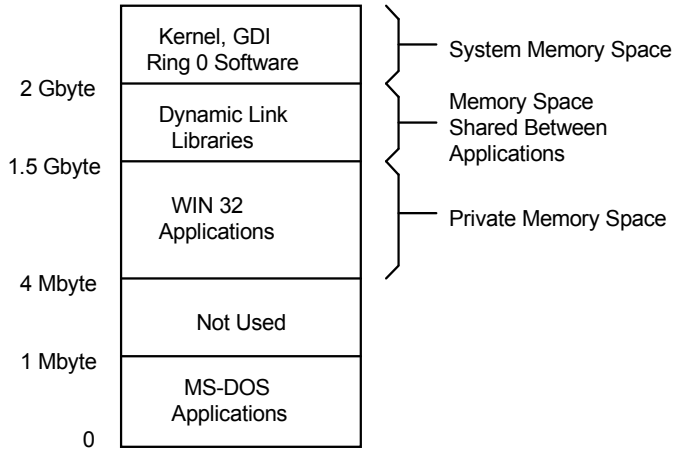
By configuring the hardware translation tables and detecting when a program moves outside an area of physical memory assigned to it, Windows 95 can make each active process "think" it has exclusive access to its own 4 Gbyte address space. Because there is rarely sufficient memory to allow all processes to have what they want, Windows 95 can make use of backing storage (hard disk) to store areas of memory not required at any time.

To use an analogy, when playing golf the only part of the golf course actually required by a player is the part containing the hole that they are currently playing. The rest of the course may as well not exist. In the same way, when an application is running not all of its program code, or data, needs to be in memory at the same time. If the hardware can keep track of what memory a program has, and alert Windows 95 when it tries to access data not actually in memory (just like our golfer moving from one hole to another) we can keep large portions of active programs on disk. Note that this works particularly well when switching from one program to another. The inactive program can be swapped out onto

hard disk. Windows 3.1 does this, but the paging mechanism is more primitive; applications were required to tell the operating system when they wanted to use particular pages. The "lock page in memory" API call alerted the Virtual Memory manager that a page must be fetched. Windows 95 does not need this, and will detect when to fetch pages automatically.

Windows 95 Program Address Space

The Windows 95 address space has the following layout:



When running in protected mode an 80386 or greater can address 4 Gbytes of memory. Not all this address space is used by applications, although you must remember that this range is substantially larger than any program ever written.

This address space is partitioned into particular areas for use by the various Windows 95 components. The bottom 1 Mbyte of memory is that set aside for use by MS-DOS applications. An MS-DOS virtual machine (VM) will run the application itself in this range.

The area of memory from 1Mbyte to 4 Mbytes is not normally used by Windows 95. This is so that WIN 32 applications can be loaded starting at address 4Mbyte, which is the same address as that used by Windows NT.

The private memory space for WIN 32 applications extends from 4 Mbyte to 1.5 Gbytes.

The area of memory between 1.5 Gbytes and 2 Gbytes is set aside as a "shared memory area". Some portions of Windows 95, for example dynamic link libraries, can be shared between several processes. The processor is allowed to make this address range available so that programs can run code in this area of memory but not change it.

Above 2 Gbytes the address space is reserved as system memory. This is where the "copy" of Windows 95 specified to the application appears to run. Of course, ring 3 applications are not allowed to touch this area, which is only accessible by ring 0, kernel mode, portions of the operating system itself.

Windows 95 and Processes

You can regard a process as an "incarnation" of an application. You start the application by selecting the icon or issuing a run command and Windows 95 creates a process to run the application and make it available to you. If you start several copies of the application, each has a process which controls the state of that "incarnation".

A process consists of:

- the program code itself, the instructions to tell the process what to do - this is the executable program file. This could be the file NOTEPAD.EXE
- an area of memory within which the process can store data. This could contain the text which NOTEPAD is currently editing.
- resources, for example if NOTEPAD has a file open the process which is running it will be assigned that file, so that other processes cannot make use of it and possible corrupt this incarnation. Other resources include things like printers and network connections.
- at least one thread of execution.

Threads and Processes

Consider a program running in memory. As the program runs, the micro-processor is fetching instructions from memory, decoding them and then executing them. Within an area of memory you can imagine a *thread* of execution as it travels along.

At any given instant a thread has reached a particular point in the program. When a thread must be suspended, for example so that another process can run, the information which characterises the thread must be stored somewhere. The information which is stored is things like the current position in memory, the contents of the processor registers, and the area of stack in use by the thread. Using this information the scheduler can re-construct the thread when it next needs to run it.

If a process is multi-threaded several threads may be executing simultaneously in different parts of the process program space - for example a multi-threaded word processor may print, spell check and edit at the same time.

A thread is started and stopped by the Windows 95 scheduler. When a program begins Windows 95 starts a thread at the entry point. A given process can start other threads which run within it, for example a spreadsheet program could start a new thread to perform a re-calculation whilst the user enters data in another part of the sheet. Note that, because only WIN 32 applications can multi-task properly, threads are only available in the WIN 32 programming environment.

Because the Windows 95 scheduling system is *pre-emptive*, threads can be stopped at any time. When a thread is stopped the scheduler will copy its current settings so that the thread can be resumed later. Note that this is in contrast with Windows 3.1, in which several programs multi-task by co-operation. In this system a single application can stop all others from running simply by not stopping.

Scheduling

Windows 95 is inherently multi-tasking. As well as user applications, components of the operating system itself rely on operating *simultaneously* with each other. The system charged with making sure that operating system components and the various applications started by the user get an appropriate share of the processor is called the scheduler.

The scheduler is a crucial Windows 95 component. Because we only have one processor, but would like to perform several tasks at the same time, the scheduler must repeatedly stop the currently executing thread and substitute another one. If it does this quickly and smoothly enough the user will be presented with the illusion of a system which is running several programs at once.

To aid the scheduler in deciding which application to run, threads can be assigned a priority level. Threads with a higher priority are **always** run before

threads with a lower one, i.e. a process at priority 6 will run all the time, excluding all processes with priority 5. However, this does not completely exclude level 5 processes from running, for example if the high priority process needs to access the disk drive, or wait for user input, it will be suspended and lower priority threads can get to run. A good trick is to have a "back burner" program which you want to run, perhaps performing a long calculation, in the machine to soak up processor time which is available when your foreground applications are waiting for other things.

Windows 95 itself contains a *null process* whose job it is to soak up processor time which nobody wants. You can get a very good idea of how busy a machine is by looking at the time spent running the null process!

Processes and Messages

Windows 95 sends processes "messages". A message refers to a particular event, such as a keypress or mouse movement. Applications work by responding to events as Windows 95 generates them.

In the WIN 32 programming model each application has its own input queue. WIN 16 applications share the same input queue. If one stops removing events, all the others are stuck This means that a single badly behaved WIN16 application can hang a windows session. WIN32 applications will not have this problem.

Process Failure and Recovery

Windows 95 is better protected against application failure than Windows 3.n.

Applications can fail in several ways:

- An application can step outside the limits set by the system, for example try to access hardware directly or address memory not allocated to it
- An application can stop responding to messages

Windows 95 will detect an attempt by a process to access areas outside its address space. In this case a General Protection (GP) fault is produced

If MS-DOS applications fail, only that application is aborted Because Windows 3.1 applications run together in a single VM the failure of one will take out all the others; however WIN 32 applications are isolated from each other

If an application is running but does not respond to events it is called a "hung" application

Hung MS-DOS applications can be killed simply by terminating their VDM. If a WIN 16 application hangs this will affect all other WIN 16 applications. In each case a "local reboot" is used to restart the affected system - Windows 95 itself will keep going

An application which has hung or caused a General Protection Fault can be "killed" cleanly. The GP fault handler runs as a separate "clean" thread and tidies up after "dead" applications. All the resources owned by an application are tagged with an ID, and so can easily be identified and returned to the system

Windows 95 Device Drivers

Both MS DOS and Windows have included the idea of Device Drivers for a very long time now.

In the case of Windows 95, a new class of driver the VxD the Virtual Anything Driver is used. Virtual Device Drivers run at ring level 0 and are written in 32 bit code. They are able to manage access to a resource and also to allow several applications to share a single resource.

Virtual Anything Drivers can also be loaded and unloaded as the system runs. This means that if devices are introduced to the system while it is up the driver can be loaded to handle that device. If the device is subsequently removed the driver can be unloaded thus * memory. Windows 95 itself uses Virtual Device Drivers at a very low level of operation and many system services are in fact provided in this way.

The Windows 95 Registry

There are currently many ways in which applications and hardware can modify the behaviour of MS-DOS and Windows 3.nn. These include CONFIG.SYS, AUTOEXEC.BAT and any one of a number of WIN.INI files. In addition most applications provide their own .INI files which can be stored in various places over the system. This makes the smooth addition and, particularly, removal of applications difficult, because you don't know what parts of the system an installation has affected.

The *registry* is the solution to this problem. It was first introduced in Windows NT, and is broadly similar in Windows 95. It is a hierarchically structured tree of data, with a root at the top and keys at various points along it. A key can contain data and other keys.

When an application saves configuration information it will put it on a particular branch in the registry. This means that the entire branch can be removed if the application is ever de-installed. The hardware settings for a particular machine are also stored as registry data, as are the settings for the individual users which a machine supports.

Registry keys can contain a number of different types of data, depending on whether you need a numeric value, bit pattern or identification string etc. They can also be protected against alteration by non-privileged users.

Registry Structure

There are seven Registry Roots:

- HKEY_CLASSES_ROOT
- HKEY_USERS
- HKEY_CURRENT_USER
- HKEY_LOCAL_MACHINE
- HKEY_PERFORMANCE_DATA
- HKEY_CURRENT_CONFIGURATION
- HKEY_DYN_DATA

Each root holds information about a particular aspect of the system. Note that the HKEY_CURRENT_USER can change depending on the username given at the start of the Windows 95 session.

When you configure a device you are changing information within the registry, which is where all settings are held. Note that you can traverse the registry tree using a program called the registry editor if you wish, but you should not do this as a matter of course.

The plug and play arbitration system will normally make changes to the registry in respect of hardware settings, but some devices are not plug and play compatible and so you will need to select the options for these yourself.

To do this you will usually use the Device Manager. This program is activated by selecting the properties of the device which you are interested in changing. Note

that because a device is an object it will have properties, which will include the hardware settings in the registry.

The properties information for a device will be composed of a number of levels, from generic information down to specific settings for particular registers on the device itself.

The Registry Editor

Should you ever have to modify the registry directly, you can run the registry editor which is in the Windows folder:

It allows you to traverse the whole registry structure, starting at any of the seven roots and moving through them. Note that you should only use this program as a weapon of last resort, in that improper modification of the registry will probably stop your system from booting correctly, in which case you will need to perform a controlled boot (see later) and ignore registry settings.

The Windows 95 File System

When creating Windows 95 Microsoft were aware of a number of limitations with the MS-DOS FAT filesystem.

They set themselves a number of design goals:

- The upgraded file system must work in 32 bit protected mode. This both improves the performance of the filesystem and improves reliability.
- The filesystem must support multi-tasking, in that several active processes may be using it at the same time. The MS-DOS system does not support this.
- The limit of 8 character filenames with a 3 character extension is a big restriction for computer users, who want to give files names which are as meaningful as possible. Windows 95 was required to support much longer names; but also to retain compatibility with programs running under the MS-DOS environment which may not be aware of these changes
- The filesystem must act as a layer within the operating system, allowing additional filesystems and network connections to be used alongside it.

All these improvements must be made whilst retaining the MS-DOS environment, so that those applications which call MS-DOS interrupts to perform filesystem functions can operate as normal.

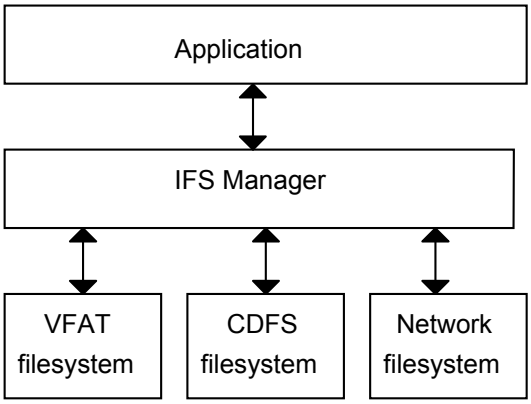
Windows 95 does this by making the filesystem a layer within the operating system. Each file system is a different version of the same layer and can be installed as required.

Installable File Systems

The key to the Windows 95 filesystem is the Installable File System Manager. It accepts messages from DOS interrupts and Windows API calls which request file services and then directs the requests to the relevant filesystem.

Provided that filesystems below the IFS conform to the requests that it will make of them, they can be added later. This means that new filesystems can be incorporated into Windows 95 very easily.

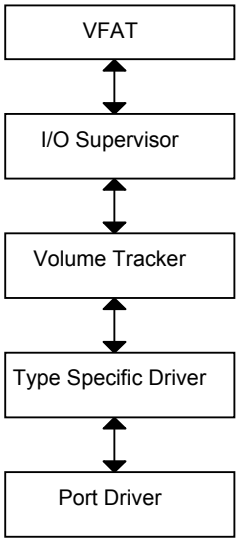
The IFS will also make requests of the *network redirector* which can attach to distant servers and make shared resources available to local applications.



The IFS manager is a VxD (Virtual anything Driver) which looks after file systems running on a Windows 95 system. It has two components, IFSHLP.SYS (which is also used in Windows 3.11 to provide 32 bit FAT access) and IFSMGR.VXD.

The IFS layer means that all filesystems are driven using the same command interface.

The VFAT Filesystem



VFAT is the 32 bit implementation of the FAT filesystem. It is implemented as a series of layers. At the top the VFAT driver converts requests into requests to the I/O Supervisor.

The I/O supervisor manages the interface between the filesystem and the actual device. It loads and registers VxD drivers for various devices as requested by the configuration manager and queues up I/O requests to the various devices. It also

provides services to allow drivers to run and sends messages to drivers as required.

The Volume Tracker is concerned with managing exchangeable media devices, for example floppy disks, memory cards, exchangeable hard disks, CD-ROM's etc. It keeps track of particular items, by writing information onto headers or recognising them, and ensures that the correct device is in the drive for a particular request.

The Type Specific Driver (TSD) is a driver for a particular type of hardware. for example hard disk versus CD ROM. It translates generic calls from the layer above into requests of the specific hardware in use.

The Port Driver is concerned with access to a particular adapter, for example a certain hard disk interface. It translates the calls from the layers above into the instructions to drive the device which is connected.

Long File Names

A major limitation of MS-DOS has always been the short filename (8 characters plus 3 character extension). This makes giving files names which reflect their contents is very difficult. This limitation has be rectified within the NT filesystem and Windows 95 also contains long filename enhancements. Unlike NT however, the Windows 95 long file names must be able to exist within the original FAT file system model, bearing in mind that many MS-DOS applications are used to reading the FAT and acting on the disk directly.

Windows 95 supports long filenames by concatenating directory entries to build up a single long name. The directory entries themselves are marked with a set of attributes which never normally occurs within MS-DOS (*read only, hidden, system file and volume*). This means that the long filename items are ignored by MS-DOS systems.

One problem remains however, old MS-DOS programs must still be able to identify files with long names. Windows 95 gets around this in the same way as Windows NT, in that two forms of filename are kept, the long form and the short form. When a long filename is entered it is parsed in a particular way and a unique short form generated. This name is then used by the standard MS-DOS file and directory manipulation interrupts, and also appears in the FAT in the usual way. This means that files with long filenames are still accessible to older applications.

For the future use of MS-DOS applications, a complete set of INT 21H functions is available for use when manipulating long filenames.

The Windows 95 filesystem goes through the procedure above to produce a short filename, for example:

a_very_long_file_named_jim.txt	->	averyl~1.txt
a_very_long_file_named_fred.txt	->	averyl~2.txt

Note that the order in which the files were processed is important, the first one we do gets the lower number.

If we had a large number of filenames which start with "a_very_long_file...." we would start seeing names like "avery~99.txt".

Windows 95 holds long filenames using UNICODE characters.

When an MS-DOS 8.3 application performs a wild card search Windows 95 uses the wild card on both the long and the short form. Matches made on sequences within the long form may cause short form filenames to be returned which do not contain the wild card string. This can be used to find particular long form names from within old applications.

Each long filename is spread over a number of directory entries, with some characters from the filename in each entry. The entries themselves contain sequence numbers so that Windows 95 can keep track of where each part fits. This means that a directory containing long filenames will be bigger that one containing short names. This can be a problem when writing at the root level of a disk, because the number of directory entries at the outermost level is restricted.

If MS-DOS applications copy and rename files using the short form of the filename Windows 95 copies the short form back into the long form where appropriate. This can result in long form information getting lost.

Some old MS-DOS programs which perform tasks like de-fragmentation and directory management may conflict with the long filename entries and cause data to be lost. For this reason, unless you have very good reason, older utilities should not be used. Windows 95 will be shipped with versions of defragmentation tools which should be adequate in any case.

In order to prevent damage to the disk surface by such programs Windows 95 will not allow the use of the direct disk write interrupts (INT 13H and INT 26H) until a particular “unlock” function has been called. You therefore have to specifically make low level access available. This unlock feature will probably be provided in the form of a program which can be run before the one which will perform low level access.

An entry in the registry can be used to switch off long filenames.

Windows 95 Networking

Windows 95 has been designed to be much more tightly integrated in terms of its Network support. Network protocols are integrated tightly into the operating system and a large number of protocols are supported. Windows 95 systems can share resources amongst themselves in the same way as Windows for Workgroups systems can operate and they can also provide services for other systems. However, Windows 95 is designed to be a Client system and not provide server based resources. The number of clients which a Windows 95 system can serve is therefore restricted to one. Windows 95 can be the client of a wide range of networking mechanisms, amongst them Novell and Microsoft Networks. The preferred server for a Windows 95 system would seem to be Windows NT.

Note that the Windows 95 networking mechanisms does not contain a large amount of security handling. We will cover security later in the course. for now it is important only to know that Windows 95 has been designed to use the Network server as the security provider.

Universal Naming Convention (UNC) Names

In order that paths to resources supplied by network servers can be expressed easily, Windows 95 makes use of the universal naming convention when expressing a path to an object on the network.

The UNC name contains the name of the server and the name of the resource which the server is sharing. A particular machine can share a resource with a particular name. These two components form the first two parts of the UNC name of the resource. After the share name a conventional path can be given to the object itself.

The object for which the path is being expressed does not necessarily have to be just a file. This convention is also used for expressing a path to a printer.

Universal Naming Convention (UNC) name:

\\server\share\path

server	name of machine sharing out
share	name of share
path	path to resource

The object might not be a file, it could be a printer or other resource.

Client-Server Network Access

All network access to resources by Windows 95 is performed on the basis of client server. One machine is the server providing the resource via the network to connected machines and the other is the client which makes use of the UNC name to identify the resource to be accessed.

The number of clients which a Windows 95 system can support is restricted to one. The positioning of Windows 95 is such that it is designed to be a client of other server systems. To this end the security of files and devices on a single windows 95 system is not particularly great instead the security is to be provided by a server system, be it Novell or Microsoft network based. In either case security validation and authentication is provided by the network server on the basis of passwords which the user will enter on the windows 95 system

Accessing Network Resources

Users can make connections to any shared resources to which they have access on the basis of security. Windows 95 links can be used to hide from the user of resource precisely whereabouts on the network that resource is located. The user simply selects the item on the desktop and windows 95 provides the underlying network access to make the resource perform the required function.

Windows 95 allows each identifiable user of the system to be given their own particular set of network connections. Furthermore the initial password given by a user to start a windows 95 session can be passed through to servers of network resources in order that the access may be validated. This means the user has to enter their password once at the beginning of the session and this is then used throughout that session to validate all network use.

Networking Protocols

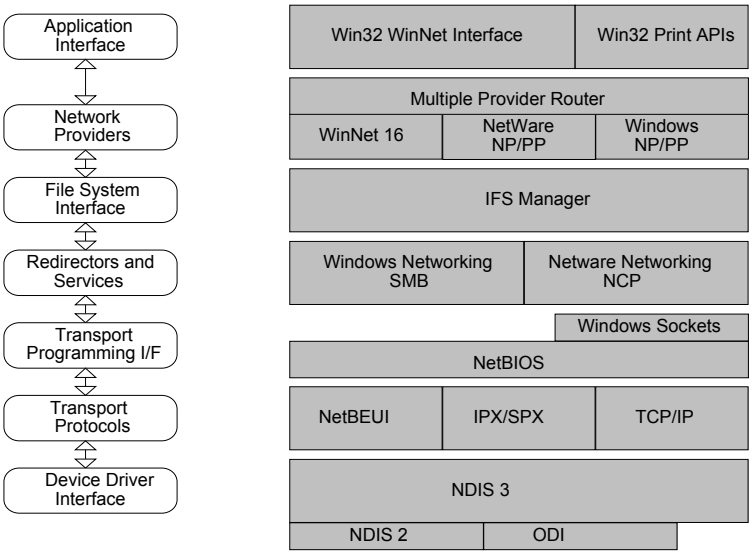
The underlying network mechanism within Windows 95 supports a large number of network protocols amongst them windows sockets, NETBIOS, remote procedure calls, named pipes and mail slots.

Windows Sockets is the standard mechanism whereby windows applications can make calls to other networked servers supporting the TCP/IP networking mechanism which is used to perform networking on UNIX hosts.

NETBIOS is the series of BIOS extensions developed to allow MS-DOS networked applications to communicate.

Remote procedure calls are a standard set by the distributed computer environment (DCE) to provide remote resources for applications running on a particular machine. Windows 95 supports both client and server RPC so a windows 95 system can both provide and access resources.

Windows 95 Network Architecture



The windows 95 network architecture is expressed as a layered protocol. Each layer in the protocol communicates with the layer above it receiving commands and returning replies and also with the layer below it sending commands and receiving replies.

The layers are loosely based on the standard layer model of the international standards organisation or open system interconnect. the use of layers within the model means that particular layers can be replaced by components which function in a different way without affecting the layers above and below. this means that the precise way the network functions is much more easy to modify.

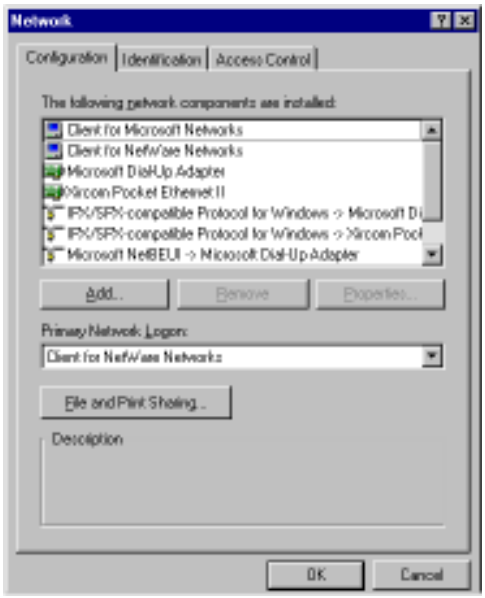
For example, applications interacting with the upper layers will not know or care that mechanisms in the lower layers are changed depending on the particular network connections in use.

Network Configuration and the Registry

Network configuration information is spread throughout the registry:

- Network Hardware
- Network Software
- Network Drivers
- Network VxDs

You should never modify this information via REGEDIT, should use the appropriate configuration or properties mechanism, which is usually via the Network icon in the Control Panel:



Windows 95 Administration

Windows 96 has been positioned in the market place as a system for use on stand-alone systems and systems linked together within a network.

Within the network Windows 96 machines can be used either as peers in a work group or as clients of a network server.

Administration of a system can be performed at either level.

When considering how to manage a workstation you should look at three option, the hardware in the system, the operating system software running on it and the applications used on the system.

All these three aspects of a Windows 96 machine are managed by use of a central storage area called the registry.

You can manage the registry directly by use of the registry editor program, but more usually set up programs for applications and property information for objects and hardware on the system will be used to change information within this registry.

Because the registry editor is a remote procedure aware application it is possible to look at and modify the registry of other machines to which you are connected via the network.

This remote administration facility is extended throughout the Windows 96 system management settings so that it is very easy for an administrator to manage systems remotely.

User Management

The administration of a system will also involve the management of the users on that system.

Windows 96 provides very powerful mechanisms for controlling who can have access to resources, setting rights and security on resources and also creating individual profiles for particular users of the system.

This means that under Windows 96 a particular user can associated with an identity which controls what they can do, the resources that they have set up and the way in which their desktop and settings appear to them.

Security

Windows 96 allows you to create an environment within which users and resources can be made secure.

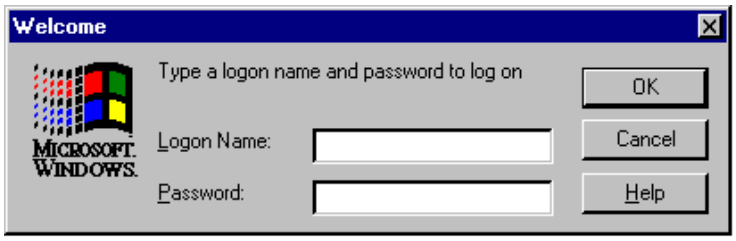
Particular resources on a machine can be protected so that only certain users can have access to them.

This can be done on the basis of machines participating in a workgroup or use can be made of security resources on server machines which can be used to manage access of resources on a Windows 96 system.

Note however that unless the protected resources are provided by a system which supports security (for example Windows NT or Novell) it is not possible to protect them from damage or unauthorised access.

User Log In

Before a user can gain access to a Windows 96 system they must pass through the log-in prompt:



It is not mandatory to log-in to gain access to resources on a Windows 96 system, but the system can be set up in such a way that without logging in very few resources can be accessed. The security regime therefore begins when the user begins using the system.

Because each user has a uniquely identifiable name, this also makes possible the customisation of the system for particular users. This makes it very easy within Windows 96 to configure a given system to behave in a different way, depending on which user is sitting at it.

User and Share Level Security

Windows 96 can make use of two different security regimes, share level security and user level security.

Within share level security passwords are assigned to particular resources. If a resources has several different levels of use, for example, at one level files can be read and at another level files can be read and written, a separate password is assigned to each level of use. this security mechanism is very easy to manage in the workgroup situation where the number of resources and users is fairly small. Any user who knows the password for a particular resource will be able to use that resource irrespective of the user name which they used to log in.

User level security makes use of a security provider to validate access to particular resources. When using user level security it is possible to assign access to particular resources on a user by user basis. This means that the user does not need to know the password to gain access to a resource. The user must instead be given rights and permissions to that resource on an individual basis.

Systems such as Windows NT or Novell servers are designed to manage a community of users. Each user is given a particular username and users can be grouped together to form particular groups who need certain rights and permissions.

The security provider deeps track of usernames, the rights and permissions which usernames have and also the groups which usernames can be members of. For a given Windows 95 system we must select the security provider to be used by that system. This provider is then accessed automatically whenever validation of an access is required.

User Profiles

If user profiles have been enabled from within the password menu users' setting information is stored in registry files. The user specific information is held in the registry file user.DAT. A copy of this file for each user. System specific registry information is held in the file system.DAT. A third file type the .POL file is used to hold system policy information. (.DAT and .POL are file extensions)

All the registry entries relating to the way a Windows 96 system appears to a user is held in a windows.dat file for that user. When the user logs in the user.dat file is used to build the registry information for the session for that particular user. Note that this means that windows applications that wish to retain setting information for particular users can write this information into the HKEY_CURRENT_USER current user key in the registry and have it stored for them automatically by the Windows 96 system.

When a user logs in Windows 96 checks for a matching profile for that user on the system. If user security is in use Windows 96 will also look on the directory on the security provider.

This means that in the user security environment a particular user can have the same profile irrespective of which system on the network they use. The environment which they maintain is simply loaded into the system whenever they access using their username.

The very first time a new user logs into a Windows 96 system, if separate user profiles are being maintained the user is then asked whether or not a profile is to be created. For systems running share level security this that the creation of new usernames and user profiles is very simple indeed requiring no additional effort on the part of the system administrator.

Of course for systems running user level security the username must be created on the security provider as well.

Mandatory Profiles

It is possible to enforce a particular set of defaults on a user by giving them a profile with the extension .MAN rather than .DAT. The profiles are loaded in the same way and the user is able to make changes to the desktop settings during a particular session. However, when the user logs out a mandatory profile is not overwritten with the new modified settings. This makes it impossible for users to make changes to their environment which last more than one session. A mandatory user profile can only be used of the profiles are stored on a network server and you are operating with user level security.

System Policies

The user profile is the area within which a user will store information relating to their desktop. A system policy is a means whereby a system administrator can determine which settings on a system the user is allowed to change. The administrator can enforce the specific settings for certain resources and allow users the ability to change other ones. The single file POLICY.POL holds the policies for all users and the system. You are allowed very precise control over those actions which a user can perform on a system with regard to modifying

their desktop environment. The POLICY.POL file can be held locally on a machine or loaded from a remote system.

You can edit policies for a local system or create a policy file for use by a number of systems and held on a central network. In either case you use the System Policy Editor.

Default Policy Settings

The default user and default computer entries in the policy file implement the default profiles for a system. By modifying these profiles you can adjust the default settings which newly created users receive. The settings for the default user and computer policies originate in the file ADMINCFG.ADM. By modifying this file you can change the way that newly installed Windows 96 systems behave.

Policy Items

There are five entries which you can change in a local user profile, control, desktop, network, shell, and system.

Control panel allows you to manage the access the user is allowed to make to system configuration information by means of the control panel utility. You can prevent access to the control panel at all or disable particular settings within it.

The desktop option allows you to specify pre-set values for desktop settings and also prevent the user from making changes to some of these settings.

Within the network setting you can prevent the user from being able to share out particular resources from this system.

The shell option allows to restrict those options available to the user from the standard Windows 96 desktop.

You can stop the display of certain items. You can stop programs from being run from the desktop and you can stop a user from being able to start up and shut down the system.

These settings are very useful when building turnkey systems for unsophisticated users. The system option allows you to restrict the users of native mode MS-DOS applications and also restrict access to registry editing tools. Such restrictions reduce the amount of damage a user can do to the system and also the ways in which a user can run programs which gain access to such a system.

Windows NT

Introduction

Windows NT is intended by Microsoft to be the operating system for more powerful personal computers and workstations. It is useful to look at because of

this, and also because it embodies many of the principles of modern operating system design.

NT Development

The origins of NT are shrouded in a history of corporate in-fighting and technological development. Whilst a full appreciation of the history is not necessary to know how the operating system works, it can help to give some background into why certain components work the way they do.

In the beginning there was the IBM PC, which ran MS-DOS. This sold very well, but as the power of the host computer rose with new processor developments and falling memory prices, people started to wonder about the limitations posed by MS-DOS.

The answer which IBM and Microsoft teamed up to produce was OS/2 - the second operating system. OS/2 was projected to be the next big thing, with huge sales guaranteed. Unfortunately, for a number of reasons - among them poor performance and late delivery, OS/2 never caught on. It was not until Windows came along that people were actually weaned away from MS-DOS.

Undaunted by the failure of OS/2, IBM and Microsoft set of to produce its successor (OS/3?). As the OS/3 work proceeded Microsoft watched the success of Windows and began to wonder if the OS/2 development was a good idea. The main problem with OS/2 was that it was designed around the 80286 chip, which had been vastly superseded by the 80386 series. After an acrimonious dispute, Microsoft ended up with the rights to the Windows developments and IBM with OS/2.

They then set out to produce their idea of the perfect operating system. The first thing they did was hire a very experienced operating system designer, who had been responsible for the DEC VAX VMS operating system.

Because some work on OS/3 had been completed at the time the NT development started, some features were left in the development but most of the NT development is "new" code.

NT was released in late 1993, to a less than rapturous reception - mainly because of the high performance required to run it and the lack of native mode (i.e. programs which run directly under NT and make use of all its features) applications.

However, Microsoft are committed to it as a central plank in their corporate strategy. This alone should ensure its success, it is likely that Windows 3.1 and Windows NT will converge into a pair of highly complimentary products, one being a restricted version of the other.

NT Workstation and Server

Windows NT is shipped in two versions. The Workstation version is for a single system which wishes to operate autonomously. Workstations can share data amongst themselves

Features of NT

Portable

Windows NT is able to run on a number of different types of micro-processor, so that Microsoft can make it available on a very wide range of machines.

Previously, most operating systems were constructed around a particular processor. Microsoft used the idea of an additional layer, between the lower levels of the operating system and the actual hardware. This layer is used to hide the precise way in which the hardware works from the operating system itself.

Furthermore most of NT is written in a high level language, so that it can be compiled down onto the required target.

Multi-Processor

You hit a limit in the performance increase you can get by using faster and faster processors. To get around this you would like to use numerous processors, but the operating system must be able to support this. NT provides *symmetric* multi-processing. Rather than additional processors being the slave of one master, which restricts performance to the maximum you can get out of the master, NT tasks are managed by any one processor, so that no single processor will limit performance.

Multiple Operating System Support

For NT system to succeed it must be able to support existing applications, so that customers are able to run their old programs when they upgrade to the new system. It does this by offering *environment subsystems* which are placed around a particular application, translating operating calls which the application makes into request to NT and converting replies in the opposite direction.

High Security

MS-DOS is inherently insecure, the operating system allows attack from malicious users and software. The reason for this is that the microprocessor on which it usually runs has no hardware mechanism for controlling access to parts of the system.

Windows NT runs on processors capable of supporting a "supervisor" mode in which the operating system runs, and a user mode for user processes. Only programs which run in supervisor mode (i.e. the operating system kernel) are allowed to access all areas of memory and physical devices. Programs running in user mode access the hardware via requests to the operating system and are limited in terms of the memory areas they are allowed to access.

In addition, NT also supports user validation, i.e. all users of NT must have a username which can be password protected. Access to all system resources can be controlled at user level, or users can be placed in groups with particular levels of access.

The way in which resources can be accessed is also very flexible, files can be marked as read only, write only or anything in between. Groups or individuals can be denied access to particular items.

The access control is backed up by very powerful logging facilities, which can be enabled or disabled as required.

Improved Filing System

The way in which MS-DOS organises files is fine for small capacity disks, but becomes inefficient when very large drives are used. NT offers a much improved filing system, with much longer file names and attribute information, as well as a faster form of directory searching based on a tree structure rather than on linear lists. The NTFS system also supports a high level of fault tolerance, transactions are logged so that if a hardware or software failure occurs the data on the disk is

not damaged, and can usually be reconstructed. NTFS also supports some RAID levels.

Layered Operation

By breaking the interface between the operating system and the underlying hardware into a series of layers you greatly increase the flexibility of the design. All of the device drivers in NT are layered in this way, so that the progression from application request to the actual tickling of the hardware passes through several well defined phases. This means that underlying portions of the service can be changed without layers higher up being aware; for example in the case of a request to access a file over the network, the application which made the request is unaware of the fact that the file access mechanism will be different, the underlying layers hide this. Things like underlying network hardware are also hidden in layers, so that at the top level you only have to worry about the function you are performing; not the mechanism by which it is performed.

Note that passing requests from one layer to another does result in a performance hit; the laying protocol detracts from the job in hand, and introduces delays. However, the benefits of doing it this way far outweigh the reduction in speed.

Object Oriented

Object oriented programming techniques are now popular for the organisation of large projects. Within Windows NT all system resources, from processes to files, are objects. All objects have a certain set of basic characteristics, such as who owns them and who can access them, along with specific characteristics depending on what they are; for example a file object will have an additional characteristic which is the data in the file itself.

All objects are managed centrally, which makes access control easy, and if additional features are required these can be added as further objects which the system will manage.

Networking

MS-DOS was not designed to work over a network; the fact that it is networked is a tribute to the ingenuity of the network developers as much as anything else. Windows NT contains networking as part of the low level design. This means that networking functions which are normally outside the operating system are actually carried out within NT. Server and Client operation, where one node makes requests of another via the network, is handled by the NT, as are Remote Procedure Calls, were an application is calling code on another machine, is also part of the NT architecture.

This means that applications are unaware of the network configuration underneath them.

NT also supports domains of users working on several machines which are networked together. A version of NT called "Server" provides the support for networked user names and services for NT workstations.

Virtual Memory

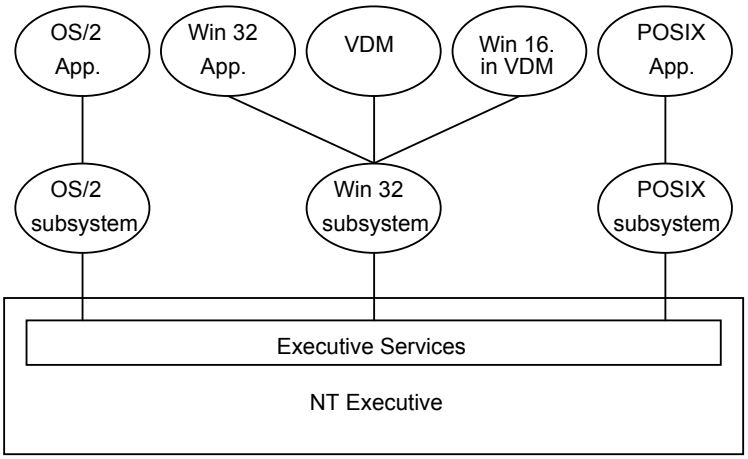
Although memory is getting cheaper all the time, programs will still want to use more than is physically present in the machine. Virtual memory is a technique where you use the hard disk to provide additional "memory" in the form of pages which are swapped in and out of real memory as required. Windows 3.1 has

some virtual memory support, but Windows NT has full virtual memory which is transparent to the application running on it.

Multi-Tasking

Windows 3.1 can co-operatively multi-task applications, i.e. it relies on an application relinquishing execution so that other programs can get control. This is not perfect, as programs are not given guaranteed access to the machine, and if an application fails the whole system crashes. Windows NT supports full pre-emptive multi-tasking, i.e. the operating system will interrupt running programs and transfer control to others. It also supports processor priority, so that the amount of execution time a given program gets can be set. Real Time priority is also available, so that particular programs can get access to the processor within certain time intervals.

NT Components



The blobs above the executive are running in user mode. The NT Executive is running in Kernel (i.e. protected) mode. Each operating system environment is supported by a subsystem which translates application requests into calls to the Executive services in NT. The Win 32 subsystem is special in that performs all the system input/output, i.e. it is the part of NT which writes on the screen and reads the keyboard and the mouse. The other subsystems access the Win 32 subsystem screen and keyboard via calls to the executive services which pass these on.

OS/2 Subsystem

The OS/2 subsystem is only available if you are running NT on a machine which has an Intel processor, which can run programs written for the 80286. The usefulness of this emulation is rather limited, as only OS/2 applications which run in character mode, i.e. no graphics, are supported.

POSIX Subsystem

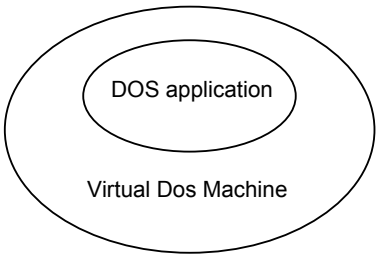
The POSIX subsystem supports applications which are designed to run on a portable version of UNIX. POSIX compliance is a popular requirement although, as for OS/2, only applications which run in character mode are supported.

WIN 32 Subsystem

This is a central component of NT. Other subsystems do not need to be running all the time, for example the OS/2 system does not need to run if no OS/2 applications are active. However, you must always have a Win 32 system because it is in charge of the screen, keyboard and mouse. When a user runs NT programs they are run as applications under the Win 32 subsystem. The logic program, which you use to gain access to the system, is run under the Win 32 subsystem.

Other subsystems communicate with Win 32 via the Executive Services layer to gain access to the keyboard and the screen.

VDM - Virtual DOS Machine



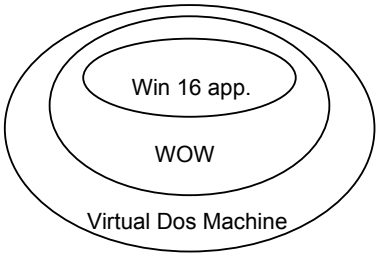
One of the things hanging off the Win 32 subsystem is a Virtual DOS Machine. This is where standard MS-DOS programs will run. All versions of NT support VDMs. If the host microprocessor is not able to run DOS programs directly, because it is not an Intel processor, the VDM is a large program which will simulate the 80286. If the host micro is Intel based the VDM simply acts as a layer between the DOS applications and Win 32.

As far as the application inside the VDM is concerned, it is making standard DOS calls and getting the usual replies. These are being converted into requests to the Win 32 subsystem for the required services.

Unfortunately; not all of the features of a real PC can be emulated easily. Any program which tries to talk directly to the hardware will cause problems for NT, which will not allow user applications to do this.

However, Microsoft hope that the VDM will allow NT purchasers to run most of their old DOS programs under NT, even if they decide to use a different processor in their workstation.

Win 16 and WOW



Win 32 is so called because the application program interface between the operating systems and programs running on it is a 32 bit one.

An *Application Program Interface* is the connection between a user program (which wants to do something) and a particular service provider (like an operating system or environment). You use an API every time you read a key from the keyboard or write to the screen from a Turbo C program, it is simply an

agreed mechanism for sending and receiving data. The original Windows API had to work on the 80286 chip, which had 16 bit data registers. It was therefore decided to pass information using 16 bit chunks and so all the API commands were implemented in this way.

Nowadays microprocessors have 32 bit registers which could be used to pass data, but this space is wasted if the API only uses 16 bit wide data. The Windows NT 32 bit API makes use of the bigger registers available on the newer machines, but is of course incompatible with the older system used by Windows 3.1. Because users will want to run old Windows applications on NT Microsoft provided another layer called Windows On Windows, or WOW. This performs conversion of 16 bit and 32 bit API calls, a process called *thunking* . The WOW layer runs inside a VDM (remember that you may want to run your old Windows programs on a different kind of processor) and lets Windows applications think they are running on a machine running Windows 3.1

Processes Under Windows NT

An application runs in NT as a *process*. A process is given an address space of 4 Gbyte, i.e. $4 * 2^{32}$. The top half of the address space is allocated to the system, the bottom half is given over to the process. Note that this memory is not actually present, but is provided by the virtual memory manager. If a process attempts to access memory outside its assigned range the operating system will take control and stop it.

Each process can have a number of executing threads within it. A thread runs within the process memory space. Threads can communicate with each other via shared memory. Threads are useful if you want to write an application which performs more than one task at once, for example you might want your word processor to perform spell checking in the background.

Windows NT Security

Before you can use NT, you must have been assigned a username and a password. NT provides comprehensive user management facilities.

Resource Protection

Resources, (files printers and the like), have *permissions*. Each permission is associated with a particular operation, for example there are write permissions, and read permissions and delete permissions for files. All resources have an owner, who can always decide on the permissions which particular items have. All resources have an *Access Control List*, which is a list of security identifications and permissions associated with them. When you try to access a resource, your security id is checked against those in the Access Control List. If you do not match any entries in the list, you are not allowed any form of access. What you are allowed to do is specified on the access token, and you are limited to this.

Security Identifiers

Each user of the system has a unique Security Identifier (SID). This is created when the user name is added to the system. The SID contains two components, one which identifies the security environment (see later) and the other which identifies the user in that environment. When you log in to a system it looks up your SID and then uses that to check against resource ACLs when you try to

access them. The idea is that the environment and the user component are completely unique.

Groups

A group is a way of collecting together a bunch of users who you want to manage as a whole. You create the group and then make particular users members of it. A group can contain as many users as you like. You could for example create a group called *payroll* and then allow it to read and write certain files. If everyone in the payroll group needs to access further items you simply allow the group to do it, and then everyone in the group can do it. As far as NT is concerned, a group is identified by its SID.

Security Environment

Particular usernames, groups, resources and SIDs exist in a *security environment*. The environment could be a single machine, or it could be a network of machines.

User Rights

Some things, for example the ability to shut down the system or change the time and date, are not manageable on a permission basis. Instead, you are either allowed to do them or not. Each user of NT has a set of rights which allow him or her to perform particular tasks on the system.

The Access Token

When you log in to an NT system as security access token is created which is then checked against all the objects that you attempt to use. The access token contains your SID and a list of SIDs of all the groups which you are a member of. It also holds a list of the rights which you have been assigned.

All objects within NT are managed by the security manager, which matches an access control list for the object against the security token of the thing trying to access it. Whenever you create a process it is assigned your token, so that it has the same rights and permissions as you have.

Network Security

Windows NT Server systems can manage groups of users from a single central server and also pass access rights between user communities which are linked together by a network.

Domains

If a set of Windows NT systems are linked together and managed as a single security environment they are said to be part of a *domain.* The domain contains particular systems and any users who wish to use these machines must have a username and password which is valid on the domain. If you do not have these, none of the machines can be used.

A master copy of the security information is held on a particular NT Server system, but other Server systems can also supply copies for validating users. This means that a very large network is not dependent on a single machine to perform all the security functions.

Trust Between Domains

If you wish to make users from one security environment able to use resources in another you can make one domain *trust* another one. If a domain trusts another, all the user names and groups which are valid in the trusted domain can be used in the trusting one. This means, for example, that users from the Manchester office could make use of resources in the London office if the London domain trusts the Manchester one. Note that for this to work you must have a Wide Area Network link between the two sites.

Index

N

O

P

R

S

T

U

V

W