

Das Boot
Howard Gilbert
(PC Lube and Tune Classic)

Getting the Boot

A computer comes with either DOS and Windows or Windows 95 preinstalled. Just turn the machine on, and they appear. Laptops will hibernate, then start up again the next day exactly where you left off. If disaster strikes and the disk is lost, reinstalling DOS and Windows is not a very complicated process. An advanced user may add Windows NT, OS/2, or Linux to the machine. With enough disk space it is possible to have all of these operating systems on the same machine. If you are in the Beta program, Microsoft will send advanced copies of various operating systems that also have to be installed somewhere. Each system comes with fairly good instructions for installing it alone, or in combination with Plain Old DOS. None of the documents tie all the systems together, and nobody ever bothers to tell you how to remove the damn thing when you have given up on it.



At some point you have to stop playing around with systems and get some real work done. Generally, this involves running some Suite of Windows programs (Microsoft Office, Lotus Smart Suite, WordPerfect, ...). All of these operating systems will run all of these programs. However, the typical Windows suite dumps up to 40 megabytes of trash in the "Windows System Directory Tree". This includes TrueType Fonts, filters, DLLs, clipart, and other junk. Then they add a zillion lines of OLE, DDE, and other stuff to WIN.INI. If you are keeping multiple copies of Windows around, one for each system, then keeping things straight can be nearly impossible.

Das Boot is The Mother of all Pre-Installation Manuals. Actually, it gets things started and leaves off at the point where the real installation manual should kick in and prove useful. It will help to clarify the initial planning. It carries things through the point where you define or select a partition. It also explains the boot sequences of multiple operating systems, the hidden files, and boot managers.

When done, we will have accumulated a collection of boots that would make Imelda Marcos envious. Click on a section to proceed (starting at the beginning is a good idea):

Hardware Initialization

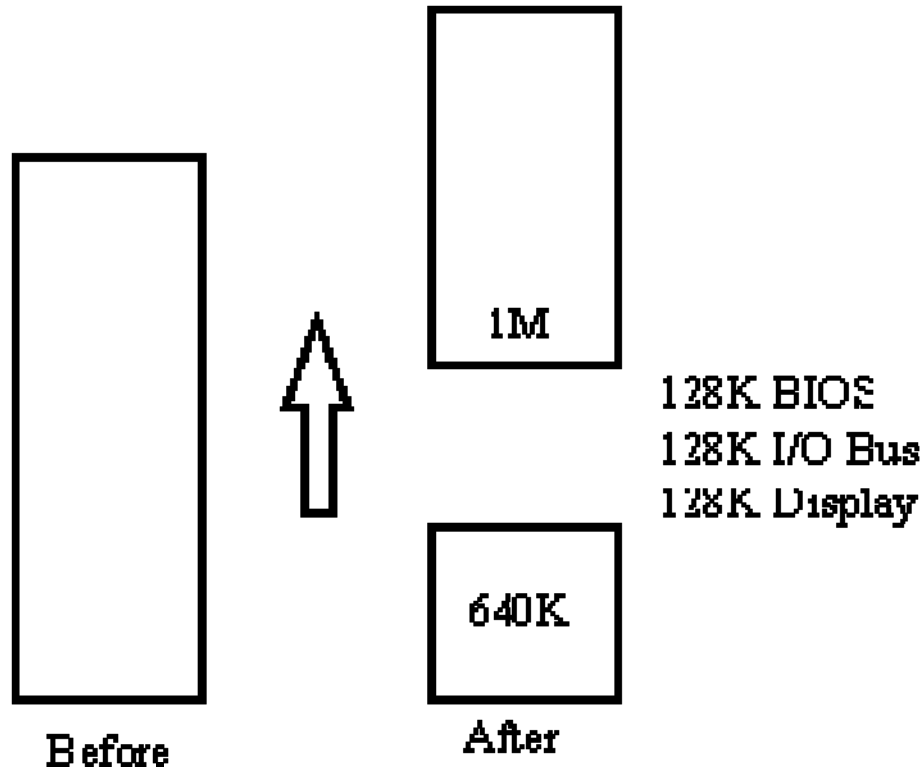
When a PC is powered on, the CPU is reset and begins executing a set of instructions stored in some kind of read-only memory (ROM) on the system board. These instructions are divided in three parts: The Power On Self Test (POST) which is run only after the power is turned on, the Initialization logic run every time the user reboots with Cntrl-Alt-Del, and the Basic I/O Support (BIOS) used by DOS to access standard devices. Traditionally, read-only memory was based on chips that were programmed at the factory. A new generation of computers contain Flash Memory that can be reprogrammed from a floppy disk. Some other systems read POST/Initialization/BIOS instructions off a special area of the hard disk.

Read-Only memory chips deliver information only one byte at a time, so they are slower than normal memory. Since the BIOS code will be used while DOS is running, most systems begin by copying the contents of ROM to an area of the standard memory chips. This reduces the amount of RAM memory that is reported as available for operating systems and applications.

8068K OK

Das Boot
Howard Gilbert
(PC Lube and Tune Classic)

The most visible consequence of the Power On Self Test (POST) for the end user is the memory test, where a number appears in the upper left corner of the screen and counts up until it reaches the total amount of memory (less the shadow area) installed on the machine. However, a number of other less visible checks are also occurring. Before anything can appear, the display adapter must be initialized. It must be set to the correct resolution and refresh rate. The hard disks must be powered up and come to operating speed. The keyboard and mouse ports are tested. If all items check out, a single "beep" is generated and the PC continues booting. Any problems are reported with multiple beeps and an error code or message on the screen.



RAM memory is installed in the machine as a row of SIMMs. Memory comes in some number of megabytes. However, the original PC architecture requires the Chip Set to open up a "gap" by logically moving memory after the 640K address to the one megabyte address. No memory is lost, the memory controller simply moves all the addresses up by 384K. The gap is divided into three 128K areas. The first area provides a "window" used to send data or commands to the display adapter. The middle 128K of addresses are mapped to adapter cards plugged into the I/O bus. The last 128K are reserved for the BIOS instructions in the shadowed ROM.

Adapter cards on the I/O bus can be configured to present an initialization program in ROM memory somewhere in the middle 128K of free addresses. In hex, these addresses are represented as C0000 to DFFFF. Each time the system is initialized, the POST program scans this area for initialization programs and runs any that it finds. This mechanism allows the display adapter to initialize itself properly (no matter which vendor or model of adapter card you own). Code on the SCSI card makes up to two SCSI disks visible and usable to DOS programs. Code on the LAN adapter will boot a diskless workstation from a LAN server.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

Hex Notation

Computers store information in bits. A bit has two values: on and off, or 0 and 1. Each time you add another bit it has two possible new values, so the number of possible total values doubles. Two bits have 4 values. Three bits have 8 values. Four bits have 16 possible values.



A long string of zeros and ones is hard to read. It is simpler to group things in packages that are easy to read and write. The first generation of computers in the early 1960's had no particular memory architecture, and it was common to cluster the bits in groups of three. This allowed the possible values to be represented from the set 0,1,2,3,4,5,6,7 ("octal notation"). As modern computers came to be built around bytes, each containing eight bits, the idea of three-bit groupings became intolerable.

A byte is more naturally divided into two halves, each with four bits. However, four bits have 16 possible values, so it is necessary to extend the ordinary set of digits. What happens next should not be a great surprise to anyone who can recognize the sequence:

2 3 4 5 6 7 8 9 10 J Q K A

Playing cards create a system of numbers with 13 values. They pick up the extra values using letters for Jack, Queen, King, and Ace. Computers create a system with 16 possible numbers, but they do it somewhat more smoothly using the sequence:

0 1 2 3 4 5 6 7 8 9 A B C D E F

This is the hexadecimal number system, commonly called hex. In this system, "A" is one more than "9" and represents the value we normally call "ten". Similarly, "B" is really eleven, and "F" is really fifteen. Just as adding one to "9" in the normal number system produces "10", so adding one to "F" in hex produces "10". The difference is that "10" in hex is the value we normally call "sixteen".

Each group of four bits can be represented by a single digit or letter. Each byte, therefore, can be represented by two such hex-digits. Hex becomes a convenient notation for representing numbers that have special meaning to the computer. For example, the one megabyte address has the value 100000. Hex values are not separated by commas. This number would usually be pronounced as "one, zero, zero, zero, zero, zero" and never as "one hundred thousand" because the latter is an ordinary decimal number. To avoid confusing hex with ordinary numbers, they are sometimes written preceded by a zero and an "x" ("0x", a convention borrowed from the C programming language). Thus 0x100000 is a way to make sure that the value is one megabyte and not one hundred thousand.

So at the end of the initialization programs supplied by ROM memory (either on the mainboard or on the adapter cards) the system has located and initialized the adapter cards and counted all the hard disks and started them spinning. Booting from a LAN Server is current beyond the scope of this document. Assume that there is at least one hard disk.

IDE and SCSI Disks

The normal boot sequence is to check first for a floppy in the A: drive, then to try and boot a system on the first hard disk. The last chance might be to boot through the LAN adapter from a server. The boot sequence can be altered with the configuration utility of some computer systems. For example, the ability to boot a floppy drive could be eliminated on public systems to improve system security.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

Whether the system boots from a floppy or a hard disk, the ROM Initialization code will read the first 512 sector from the device. This contains a program which is then allowed to continue executing. On a floppy diskette, the boot sector program is written when the diskette is formatted. Each operating system has its own version of the boot sector program. When DOS is told to format a floppy, but not to put a copy of the DOS system files on the volume, it writes a boot sector program that produces the frequently seen message:

**This is not a system disk
Press any key to restart**

If there is no floppy in the A: drive, the ROM initialization logic is supposed to load the first sector from the first hard disk. However, if there are several hard disks installed using several disk controllers, it may not be obvious just which disk comes first. Most desktop PC's come with built-in support for IDE disks, though a SCSI disk controller can be used as a replacement or to provide supplemental storage. These two disk architectures should be explained.

The IDE option is the most widely used and least expensive. The IDE disk interface is based on the ISA I/O bus architecture used on most desktop machines. Because it essentially duplicates and extends the native bus, the control logic for an IDE system is minimal. IDE disks also duplicate the old ST506 command set of the first hard disk controller (the Seagate Technologies 506) installed in IBM PC's. The ST506 control functions are directly supported by the ROM BIOS, by all versions of DOS, and by all other systems.

An IDE interface supports one or two hard disks. Over the last year, computers shipped with a EIDE interface that supports two cables each able to connect two devices for a total of 4 devices. Most desktop units ship with one IDE or EIDE hard disk, and that is usually the boot device.

Because of differences between the original IBM BIOS and the ST506 hardware, older systems were limited to IDE disks with no more than 528 megabytes. A new system has an improved BIOS that lifts this limit. A new operating system like Windows 95 has support that bypasses the BIOS. Even with Plain Old DOS, you can usually install a driver that fixes things. However, you can only use operating system drivers after the operating system boots. The Boot process itself uses the BIOS. It is generally a good idea to make sure that the C: drive, and any other bootable partition, begins and ends within the first 528 megabytes of the first IDE disk.

IDE is a logical extension of the ISA bus. It is not found on other systems (Macintosh, RISC workstations, high end servers) that use other I/O buses. IDE is also designed to handle only one I/O operation at a time. Servers running a multitasking operating system (OS/2, NT, Netware) will want to run several I/O operations to multiple disks concurrently. IDE is therefore limited to single user desktop systems based on PC architecture. All the other (more expensive) machines use SCSI I/O.

SCSI is a more general standard for the attachment of all sorts of devices. Disks, backup tape units, CD-ROM, high-capacity optical systems, and a variety of other devices support SCSI attachment. SCSI devices can be located inside the PC (connected to the adapter card by a flat ribbon cable) or external (connected by a cable similar in size to the parallel printer cable).

If a system is to use the advanced features of SCSI (high performance, multitasking, various device types) then it needs special support. SCSI adapter cards are made by several vendors (Adaptec, Future Domain, and IBM for example) and advanced operating systems (OS/2, Windows NT, Unix)

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

provide extensive support for each type of card. However, this support along with the rest of the operating system must be loaded off disk, and if the only disk available is a SCSI disk, then there is a clear chicken-egg problem. How do you load the SCSI adapter support from a disk controlled by the SCSI adapter?

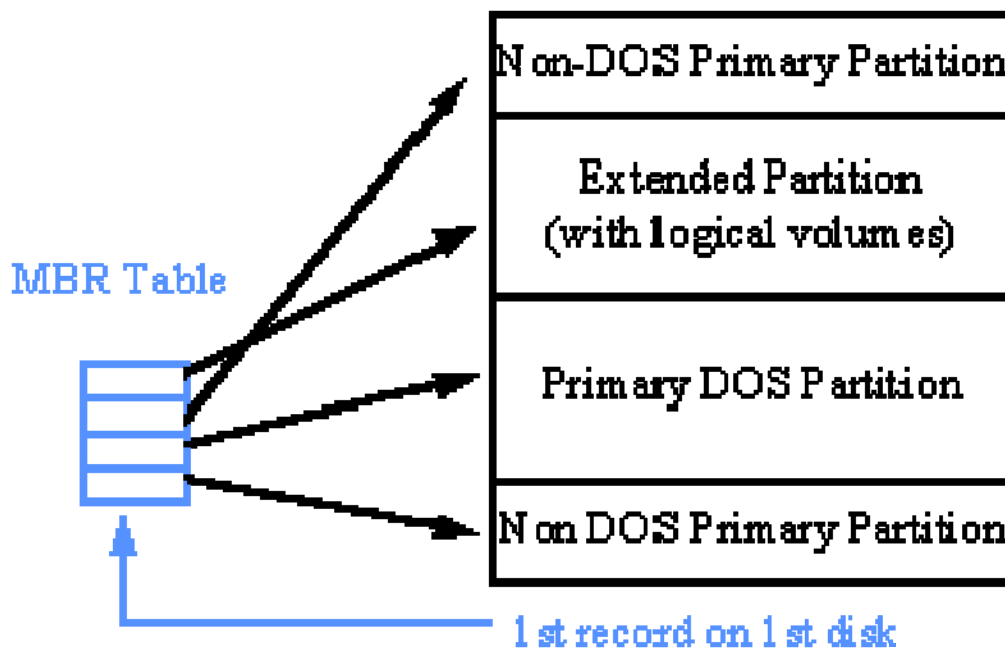
In response, the SCSI adapter cards for a PC contain extra logic to convert the first two SCSI disks into simulated ST506 disks that can be used by the standard BIOS. An operating system and its device drivers are loaded using the ST506 simulation, then the system switches over and uses the more advanced support.

If a system has only IDE drives, then the first IDE disk is the boot device. If a system has only SCSI drives, then the ROM initialization logic on the first SCSI adapter card detects this situation and inserts the first SCSI disk (under ST506 simulation) as the boot device. If both IDE and SCSI disks are present, the SCSI controller defers to the IDE support and the system boots from the first IDE disk.

Partitions and Volumes

Having identified the first hard disk, the ROM initialization code now reads the first record from that disk into storage. This is the Master Boot Record or MBR. The first 446 bytes of this record are a program that is to be executed. The rest of the record is a two byte header and four 16-byte table entries to define up to four partitions. These can be four Primary Partitions or three Primary Partitions and one Extended Partition containing Logical Volumes.

The MBR has become somewhat famous because of a number of "boot sector virus" programs that replace the MBR with another program that damages your system. Developing a program to detect



MBR changes can be difficult because there is no universally accepted version of the MBR program. In ancient times, vendors with a special disk architecture would supply their own disk format program and write an MBR that handled the peculiarities of the disk. Advances in technology should have

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

solved that problem, and today all IDE and SCSI disks are fairly standard. However, IBM's OS/2 and Microsoft's Windows NT develop different and sometimes conflicting new MBR features.

A proper MBR will look at the partition table and choose the Primary Partition that is marked Active (a DOS word) or Startable (an OS/2 word for the same thing). The Extended Partition is never startable, and the MBR doesn't even know about the Logical Volumes contained in it. The MBR program reads the first 512 byte sector from the Startable Primary Partition into storage. It then jumps to the program in that sector.

The reader may experience *deja vu* at this point. The ROM Initialization logic read the 512 byte sector at the beginning of the hard disk to load the MBR. The MBR then reads the first 512 byte sector from the beginning of the active partition to bring in an operating system loader program.

Without some non-standard extension, the MBR only supports four disk partitions. Furthermore, some IBM systems (particularly recent PS/2 systems) use one of the entries for a System Partition containing the utility to configure the system and I/O bus (what used to be the "Reference Diskette"). This is not a serious problem, because DOS traditionally has only supported two partitions.

The first primary partition that contains a DOS file structure (a FAT file system) is called the Primary DOS Partition for that disk. If it is on the first hard disk, it becomes the C: drive. Microsoft is partial to the C: drive and requires that DOS, Windows 95, and Windows NT store their bootable system files on this drive. The three Microsoft operating systems always start by loading files off the C drive. Of course the \WINDOWS or \WINNT35 directories can be on another disk letter. However, a couple of megabytes of hidden files in the root of the C: drive are used to start loading DOS, Windows 95, or NT. One of the four partitions on the disk can be the Extended Partition. It can be subdivided into logical volumes. Each logical volume is assigned a disk letter and can be formatted with a separate file system. There is no limit on the number of logical volumes other than the fact that there are only 26 letters in the alphabet and A, B, and C have been taken.

The MBR identifies the location and size of the Extended Partition, but the MBR has no information about the logical volumes. Neither the Extended Partition nor any of its volumes can be marked Active or Startable. An operating system can be installed on a logical volume, but it cannot be booted directly from the MBR. The trick is then to install a boot sector and some sort of Loader utility on one of the three Primary Partitions. The MBR then boots the Loader utility. The Loader utility can have information about the logical volumes and find an operating system stored in them.

OS/2 has Boot Manager. IBM tried to give this the abbreviation MOST, but like it or not, Boot Manager is unfortunately going to be shortened to BM. The OS/2 FDISK utility installs BM in a separate minimal (1 meg) non-DOS Primary Partition somewhere on the first disk. BM takes up one of the four entries in the MBR. It does not have a DOS file system, and is not assigned a disk letter. But it can be marked active/startable, and when it is the MBR loads the Boot Manager code into memory and runs it.

Boot Manager can load an operating system from any partition or logical volume on any of the first two disks. BM is not a full operating system, and it does not have advanced support for specific SCSI adapters. So it is limited to disks accessible through the ST506 emulation. Since most SCSI adapters only provide BIOS mappings for the first two hard disks on the SCSI bus (and frequently only for devices with ID 0 and 1), it may not be possible to use BM to load an operating system off a third SCSI disk. One should at least examine the specifications of the SCSI adapter card before planning to install any operating systems on a third disk. The BIOS on older systems with IDE (instead of EIDE)

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

disks may only support the first 528 megabytes on each disk. BM itself would have to be contained within this area, and any operating system that it was supposed to start would have to be on a partition or logical volume within the first 528 megs of the disk.

Guess what Boot Manager does to load an operating system from another partition or logical volume? Right! It reads the first 512 byte sector from that partition or volume into storage and runs whatever program is contained there. To recap, the ROM Initialization code reads the first 512 byte sector off the first hard disk (the MBR) and runs the code in it. The MBR then reads the first 512 byte sector from the active partition and runs the code in it. If the active partition is the Boot Manager, it then reads the first 512 byte sector from some other partition or logical volume and runs the code in it. With some luck, the process stops and you have found a real operating system.

Boot Manager cannot be installed on the second or third hard disk, because the ROM will not load an MBR from anything except the first disk. Until BM is installed on the first hard disk, the OS/2 installation process will not allow any partition other than the C: drive to be selected as the target for the operating system. After installing BM somewhere on the first disk, then the OS/2 installation process will allow any other logical drive to be marked "installable". If Boot Manager is installed on an IDE disk, it should be in the first 528 megabytes to ensure that the MBR and BIOS can find it.

Boot Manager is installed onto the disk by the OS/2 FDISK utility. It is also configured by attributes that FDISK assigns to the various partitions. BM and FDISK are dependent on the structure of the MBR. Unfortunately, change made to the MBR by Windows NT or Chicago can make BM or OS/2 FDISK "ill". Initially, FDISK may refuse to mark a partition installable or startable. Eventually, BM may refuse to start.

Partitions and volumes are managed by a system utility. In DOS and OS/2 the utility is called FDISK. In Windows NT it is the Disk Administrator. There are also a number of separate utilities that manage partitions on behalf of Unix-like operating systems. The MBR logic described above places certain restrictions on all disk management utilities, but some are more flexible than others.

OS/2 doesn't seem to recover well when BM is damaged by changes to the MBR made by Microsoft installation utilities. It is important, however, to realize that BM contains almost no real information. It simply assigns a name to all of the partitions that contain bootable operating systems. When BM or OS/2 FDISK start refusing requests to change partitions or parameters, the best response is to delete the BM partition entirely (using either OS/2 or DOS). Then reboot the two OS/2 installation diskettes, create a new BM partition, and reenter the names of the operating systems on any other partitions.

The order of the Partition definition entries in the MBR does not necessarily correspond to their physical location on the hard disk. The Extended Partition (if it exists) is always in the fourth and last MBR table entry. When DOS searches for its Primary DOS Partition, it searches the table entries in the order in which they appear in the MBR, not the order in which they are arranged on disk. One trick used by some disk management utilities is to create two alternate versions of the DOS C: drive on two Primary Partitions of the same hard disk. DOS will accept the one which is defined in the first MBR table entry. By swapping the two entries, alternate versions of DOS can be booted. Neither DOS system can access files in the other system partition, because DOS does not assign any disk letter or provide access to other primary partitions.

Some disk management utilities require that you create the Extended Partition on some part of the disk, then subdivide it into logical volumes. However, the MBR algorithm can be more flexible. The Extended Partition must occupy one area of the hard disk. While it is defined by the last entry in the

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

MBR table, it does not have to occupy the last area of space on the disk. Some of the FDISK-like utilities allow the Extended Partition to grow or shrink as new logical volumes are created from free space immediately in front or behind its previous area on disk. The real restriction is that a Primary Partition cannot be placed between two logical volumes, because that would imply that a Primary Partition overlaps the Extended Partitions.

NT allows disk letters to be arbitrarily assigned to disk volumes. DOS and OS/2, however, assign disk letters at boot time to the partitions that exist. First, a disk letter is assigned to the first Primary Partition with a DOS "FAT" file system found on each disk. Then disk letters are assigned to all the logical volumes on the first disk, then to logical volumes on the second disk, and so on until the system either runs out of disks and volumes or runs out of letters.

A user has an IDE hard disk divided into a Primary DOS Partition (C:) and a logical volume (D:). She then goes to Diskorama and buys a second IDE disk, installs it in the machine, and creates another Primary DOS Partition and another logical volume. After rebooting, the Primary DOS Partition on the new disk has become D: while the old logical volume has changed to the E: drive. The DOS and OS/2 algorithm assigns letters first to all the Primary Partitions on all the disks. This will, in general, mess up all the disk letters stored in INI files and other configuration data. To avoid trouble, the second disk should be configured with no Primary Partition. It should only have an Extended Partition with two logical volumes. Then the old disk will continue to have drive letters C: and D: and the new disk will get letters E: and F: as was probably intended.

File Systems (FAT, HPFS, NTFS)

At the BIOS level, a disk partition contains sectors numbered 0, 1, etc. Without additional support, each partition would be one large dataset. Operating systems add a directory structure to break the partition up into smaller files, assign names to each file, and manage the free space available to create new files.

The directory structure and methods for organizing a partition is called a File System. Different File Systems reflect different operating system requirements or different performance assumptions. Unix, for example, has the convention that lowercase and uppercase are different in file names, so "sample.txt" and "Sample.txt" are two different files. DOS and the systems that descend from it (Windows 95, OS/2, and Windows NT) ignore case differences when finding file names. Some File Systems work better on small machines, others work better on large servers.

Each partition is assigned a type (in the MBR for primary partitions, in the Extended Partition directory for logical volumes). When the partition is formatted with a particular File System, the partition type will be updated to reflect this choice.

The same hard disk can have partitions with File Systems belonging to DOS, OS/2, NT, and Linux (or other Unix clones). Generally, an operating system will ignore partitions whose type ID represents an unknown file system type. It is fairly easy (given a big enough disk) to install all of the different operating systems and all of the File System types. There are a few rules to make things simple.

Each File System is described in detail in a separate section.

FAT File System

The FAT File system is used by DOS and is supported by all the other operating systems. It is simple, reliable, and uses little storage.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

VFAT

VFAT is an alternate use of the FAT file system available in Windows 95 and Windows NT 3.5. It allows files to have longer names than the "8.3" convention adopted by DOS. VFAT stores extra information in the directory that older DOS and OS/2 systems can ignore.

HPFS

HPFS is used by OS/2 and is supported by Windows NT. It provides better performance than FAT on larger disk volumes and supports long file names. However, it requires more memory than FAT and may not be a reasonable choice on systems with only 8 megs of RAM.

NTFS

NTFS provides everything. It supports long file names, large volumes, data security, and universal file sharing. A departmental NT file server will probably have all its partitions formatted for NTFS. Because the other operating systems cannot use it, NTFS is less attractive on personal desktop workstations or portables.

File Systems and Disk Letters

DOS and Windows 95 can only boot from the C: disk. Technically, the C: letter will be assigned to the first Primary Partition on the first hard disk that has a FAT file system. In no case can DOS boot from a second hard disk or a logical volume in the extended partition. However, if as the system comes up, the DOS boot sector and DOS files turn out to be on the second Primary Partition on the first hard disk, then this will not be a problem so long as the first partition has a non-FAT file system. DOS simply ignores primary partitions that are formatted for other operating systems.

Some people exploit this feature. They put an HPFS or NTFS file system on the first Primary Partition, and a FAT file system on the second. This can produce confusion. When the other operating system boots up, it will now assign letter C: to its first partition, and the disk that DOS calls "C:" will become "D:" on the other system. If the two systems share application programs, it becomes very difficult to configure INI files as the drive letter keeps changing back and forth. It is a simpler and safer strategy to accept the view that the first Primary Partition on the first hard disk should be formatted with the FAT file system and should be the C: drive in every operating system.

Choosing a File System

The performance problems with FAT have been greatly reduced by various strategies to use Cache memory and to periodically DEFRAG the disk. FAT is the only system fully supported by DOS and Windows 95. It is also a perfectly acceptable choice under Windows NT and OS/2. FAT systems require the least memory and are the best choice on small machines.

Although it is simpler to manage a few larger volumes, FAT performance degrades with volume size. The distance between the directory and the data increases the disk movement, and larger allocation units waste space. A good rule of thumb would limit FAT volumes to a maximum of 255 megabytes.

FAT proven to be quite reliable and is fairly immune to damage. When the system crashes, FAT can "misplace" disk space that was being allocated to a file. CHKDSK (or Microsoft's newer SCANDISK) will recover the missing space. Less frequently a really serious error could leave the same sector of disk space assigned to two different files. Such "crosslinked" files are damaged, and once this occurs the entire volume is suspect. The preferred recovery would be to back everything up, reformat the volume, and restore the data. Crosslinked files could be produced by a damaged operating system, or by a hardware problem in the disk subsystem itself.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

HPFS is supported by OS/2 and Windows NT. Although it is not officially supported by DOS or Windows 95, there are shareware drivers (such as AMOS3) that can provide these systems with at least Read-Only access to HPFS files. Since OS/2 does not support VFAT, it cannot use long file names on a FAT volume. Many OS/2 software packages require long file names. An OS/2 system with enough memory and disk space should have at least one HPFS volume to support such packages.

Only Windows NT can use data on an NTFS volume. NTFS is required to provide full security on an NT File Server, and to support Macintosh datasets. On desktop workstations that run other operating systems as well as NT, NTFS is probably more trouble than it is worth.

A good general principle is to put FAT volumes first on a disk, then HPFS, and finally NTFS. All the systems will see the FAT volumes and will assign them disk letters. With device drivers for DOS, all the system will see the HPFS volumes as well. The NTFS volumes will only be available to Windows NT and will be ignored by the other systems.

Planning Windows Applications

There are two types of Windows programs. A program designed for Windows 3.x is called a WIN16 application. It can also run on Windows 95, Windows NT, OS/2, and under the Windows emulator of Macintosh and some Unix systems. A newer program will be written to the WIN32 interface. Such a program currently runs only under Windows 95 and Windows NT.

A typical Suite of office applications will take up a lot of disk space. Even on today's large disks, there isn't enough room to hold several operating systems and duplicate copies of all the applications. The WIN16 programs will run on the new operating system, but as WIN32 versions of the same programs come out they add additional features that are hard to ignore. Once you become comfortable with the new systems, it is easier to upgrade to all WIN32 support.

Windows NT was designed to install in the SYSTEM32 subdirectory of C:\WINDOWS and to coexist with an existing Windows 3.x system. However, Windows 95 and Windows NT cannot share directories or files. Each must be installed separately.

WIN16 programs were generally configured using *.INI files that were stored in the WINDOWS directory. Not only do the new operating systems have different versions of the WINDOWS directory, they also support configuration of WIN32 programs in a common database called the Registry. Registry entries can only be made to the currently running system. When a machine supports several operating systems, WIN32 programs must be reinstalled on all of the systems.

This is not as bad as it seems. The second time that SETUP is run, it can be pointed to the directory used in the previous installation. It will skip all the files that are already installed and only add files to the system directory and entries to the current Registry.

Applications that are upgraded to WIN32 will no longer run under OS/2. One can keep around the old WIN16 versions of some programs, use only native OS/2 programs under OS/2, or discard OS/2 all together. IBM knew this was coming and decided to stonewall.

DOS System Files

The modern DOS operating system is distributed on 3-5 high density floppy disks. It comes with backup utilities and (depending on how the lawyers feel) disk compression drivers. However, all the

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

stuff that goes into C:\DOS and its subdirectories are programs and utilities. The core DOS operating system consists of six files:

- The boot sector is a 512 byte record placed at the beginning of the C: drive when DOS was installed, or placed there subsequently using the "sys c:" command.
- Two "hidden" files are stored in the root directory of the C: drive. They do not show up in a DIR listing unless the /A switch is used. On IBM PC DOS systems, they are IBMBIO.SYS and IBMDOS .SYS. On MS DOS systems, they are called IO.SYS and MSDOS.SYS. These files form the kernel of the DOS system.
- COMMAND.COM is the "shell" or command interpreter. It prints out the "C:\> " prompt and reads user commands. It also supports BAT files.
- The user configuration files are CONFIG.SYS and AUTOEXEC.BAT. The reader is assumed to be familiar with these files.

COMMAND.COM is initially stored in the C:\ root directory. The problem is that OS/2 and Windows NT have their own versions of COMMAND.COM. To avoid confusion, each COMMAND.COM should be stored in the subdirectory that belongs to its particular operating system. In normal use, this means that the DOS version should be in C:\DOS. To relocate it, two statements must be added to the user configuration files:

```
"SHELL=C:\DOS\COMMAND.COM" is added to CONFIG.SYS.  
"SET COMSPEC=C:\DOS\COMMAND.COM" is added to AUTOEXEC.BAT.
```

The hidden files IBMBIO, IBMDOS, IO, or MSDOS have names that do not conflict with each other or with any system file belonging to any other operating system. They can stay in the C:\ root directory no matter what gets added to the system. This means that the volatile part of the DOS system consists of the boot record, C:\CONFIG.SYS, and C:\AUTOEXEC.BAT.

Putting DOS on a Floppy

To install a copy of the DOS system files on a floppy, simply issue the command:

```
FORMAT A: /S
```

Recovering Damaged DOS System Files

If the DOS system files have been clobbered, boot the same release of DOS from a floppy. When using the Install Diskettes for DOS, wait till the first panel appears and then escape from the installation system to the command prompt. Type

```
SYS C:
```

Planning for OS/2

OS/2 comes preinstalled on a bare system when you buy some configurations directly from IBM. Normally, however, systems are shipped with DOS and Windows preinstalled. The most cost effective upgrade is then to buy OS/2 Warp and add it to the system.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

Dual Boot

If OS/2 is installed onto the C: disk along with an existing DOS system, it will establish a Dual Boot configuration. DOS files will be left in the \DOS directory. The bulk of the OS/2 system will be installed in the \OS2 directory. A few additional utility directories may be created (DESKTOP, SPOOL, NOWHERE, PSFONTS).

Dual boot manages the volatile files in the root directory. In particular, it switches between DOS and OS/2 versions of:

- the boot sector
- CONFIG.SYS
- AUTOEXEC.BAT

During OS/2 installation, the DOS versions of these files are copied to the C:\OS2\SYSTEM directory under the names BOOT.DOS, CONFIG.DOS, and AUTOEXEC.DOS. OS/2 then installs its own boot sector, CONFIG, and AUTOEXEC files in the C:\ root.

Later on, the C:\OS2\BOOT.COM utility can be used to switch between the two operating systems. From OS/2, one switches to DOS with the command:

```
boot /dos
```

While from DOS, one switches back using the command:

```
\os2\boot /os2
```

Note that the BOOT program is not normally in the DOS path, so it may be necessary to fully qualify the name as \os2\boot so the system can find it.

In either case, the boot utility copies the boot sector record, CONFIG, and AUTOEXEC files of the current operating system to the \OS2\SYSTEM subdirectory, then it restores the saved copies of these files from the other system. When DOS is running, \OS2\SYSTEM has saved files named BOOT.OS2, CONFIG.OS2 and AUTOEXEC.OS2. When OS/2 is running, the *.DOS versions of these files are saved.

The OS/2 system also has a number of hidden system files in the C:\ root directory. They have names that will not cause any conflict with other operating systems.

Boot Manager

The Boot Manager utility is installed in its own non-DOS 1 megabyte Primary Partition on the first hard disk. It may be installed anywhere on a SCSI disk. On an IDE disk, it is safest to put Boot Manager somewhere in the first 528 megabytes where the BIOS is certain to be able to find it. The partition into which OS/2 is installed should also be completely contained within the first 528 megs. Newer systems may have a more advanced BIOS that eliminates this restriction. When the Boot Manager is installed it is marked Active/Startable. The installation of another operating system, such as Windows NT, may temporarily disable the Boot Manager. When the installation is done, just mark it Active/Startable again using some version of FDISK/Disk Administration.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

The ROM initialization code goes to the Master Boot Record. The MBR sees that the Boot Manager is active, so it loads that program into storage. Boot Manager presents a menu on the screen, and lets the user make a selection or boots the default system if there is no response within a time-out period.

Boot Manager loads another system by reading the first 512 byte sector from the partition or logical volume into storage. Boot Manager can load any system from any volume on any disks, provided that the system is smart enough to initialize properly. OS/2 can be installed on any disk letter and will load under BM.

With Boot Manager, the DOS system can remain on the C: drive. The \OS2 directory and its associated junk (DESKTOP, SPOOL, PSFONTS) are loaded on another volume. The user then selects either DOS or OS/2 from the Boot Manager menu.

It is not necessary to juggle the Boot Sector records. In this configuration, the DOS boot sector is always the first 512 bytes of the C: drive, and the OS/2 boot sector is always the first 512 bytes of another partition or logical volume. The Boot Manager selects a volume and then loads the boot sector from that volume.

Recreating Boot Manager

The Boot Manager utility is created with the OS/2 FDISK program. The BM partition contains only a minimum amount of information needed to create the menu. If the user is familiar with the system layout, it is fairly simple to delete the Boot Manager and then recreate the partition.

This is necessary because the Boot Manager partition is occasionally corrupted. The easiest way to mess it up is to use Windows NT Disk Administrator or Chicago FDISK, but there is no single bug to avoid. It does not always happen, but it occurs sometimes. The symptom is that the Boot Manager won't run, OS/2 cannot be started, and the OS/2 Installation Diskettes hang when you try to boot them because they recognize the Boot Manager partition but get confused by its contents.

To solve this problem, boot DOS from a floppy. The DOS 6.x FDISK utility may not know about Boot Manager, but it knows that something is in a 1-megabyte non-DOS partition at the end of the disk. Tell DOS FDISK to delete the 1 megabyte non-DOS partition. Now the OS/2 install diskettes will not see a corrupted BM partition. They will load successfully. It will then be possible to run the OS/2 FDISK from the Install floppies to reinstall Boot Manager and reconfigure the menu.

Dual Boot or Boot Manager?

Most people get their DOS and Windows preinstalled on the hard disk by the system vendor. Most system vendors allocate one big C: partition on the hard disk. This is not a good idea, but they do it anyway.

In order to use Boot Manager and to install OS/2 on another disk letter, it would be necessary to scrub everything off the hard disk, repartition it, and then reinstall all the software. This is too much trouble, so users tend to be lazy and just accept the large C: volume. In this environment, the only option is to use Dual Boot. OS/2 can then be installed in the existing C: volume and will coexist with DOS.

What's wrong with this. Suppose the system comes with DOS 6.1 and you add OS/2 Warp. Now IBM gives away a set of disks to upgrade DOS 6.1 to DOS 6.3. The user is supposed to be smart enough to switch back from DOS to OS/2 before installing the upgrade. Unfortunately, this little step gets overlooked. So the DOS 6.3 Upgrade process clobbers the OS/2 boot sector and probably trashes the OS/2 version of CONFIG.SYS. Assuming that CONFIG.SYS is backed up somewhere, the problems

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

can be overcome. However, it is generally an invitation to problems to have two operating systems on the same disk volume with different versions of the same named file.

Boot Manager can put DOS on the C: drive and OS/2 on the D: drive. Upgrades to DOS go to the DOS volume and do not effect the OS/2 system. The system can still get screwed up, but it takes more work to trash things. Boot Manager is the safer and more general solution if you can take the time to repartition the disk.

How to Recover a Trashed Dual Boot

If someone installs DOS onto a Dual Boot C: drive and clobbers OS/2, it is possible to recover the system without reinstalling. Normally, when DOS is running the OS/2 root files are stored in \OS2\SYSTEM under the names BOOT.OS2, CONFIG.OS2, and AUTOEXEC.OS2. When OS/2 is running, the DOS versions of the files are stored in the same place as BOOT.DOS, CONFIG.DOS, and AUTOEXEC.DOS. However, in this configuration BOOT.OS2 is not deleted. It may, however, be marked as "Hidden", so it will only show up in a DIR listing using the /A switch.

So if DOS is installed on top of OS/2, the problem is that the C:\ root has DOS files and the C:\OS2\SYSTEM directory also has DOS files. Assuming that the OS/2 versions of CONFIG and AUTOEXEC have been saved somewhere, the trick is to copy them back into \OS2\SYSTEM as CONFIG.OS2 and AUTOEXEC.OS2. Then unhide the BOOT.OS2 that is already in the directory. Now you have a saved OS/2 system ready to be restored, and can use the "boot /os2" command to switch systems again.

Putting OS/2 on a Floppy

An IBM employee has written a package to build bootable diskettes from you OS/2 system. It is called BOOT2X and is available on various bulletin board systems or from Almaden or Watson.

Can OS/2 be Completely Removed?

The short answer is "Only by reformatting all the volumes" but in practical terms it is not quite that bad. The OS/2 system itself consists of:

- The boot sector record
- Hidden files named OS2* in the root of the disk
- The \OS2 directory and its subdirectories
- The OS/2 versions of CONFIG.SYS, AUTOEXEC.BAT, and STARTUP.COM
- The Desktop directory and its subdirectories (for Workplace Shell definitions)
- \SPOOL (print job) and PSFONTS (fonts)

The boot sector can be overwritten and the other files and directories can be deleted. If these files are deleted while running a copy of OS/2 (say by booting off the Install diskettes and then deleting the files) then the Extended Attributes for the deleted files will be removed. If OS/2 was installed in a FAT partition, then these directories can be deleted under DOS. However, DOS is not smart enough to delete the Extended Attributes as it deletes the files and directories.

Before installing OS/2 on an old disk, it is a good idea to run the OS/2 CHKDSK program to clean up any problems. Rather than relying on the standard script, load the Install diskettes and run the program manually. The new OS/2 will inherit Extended Attributes defined for any of the remaining directories and files, but since they belong to applications they will not effect the correct function of a subsequent operating system.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

Planning for Windows NT

The Bulk of the Windows NT system is installed in the \SYSTEM32 subdirectory of either C:\WINDOWS or x:\WINNT (on any disk volume). However, the NT boot process always starts with files installed on the C:\ drive.

On any Personal Computer, the initial stages of the Boot process are handled by the BIOS. Windows NT is a portable operating system that runs on RISC CPU chips like the DEC Alpha or the PowerPC. RISC systems don't have the same memory structure or BIOS services as an Intel based PC. To accommodate this, RISC computer vendors have adopted a standard called ARC (Advanced RISC Computer). The NT Boot process is designed to accommodate both ARC and Intel machines. However, since most people will not buy a RISC system, this article discusses only the Intel boot process.

Most computers come with DOS or Windows 95 preinstalled. Unless NT is told to reformat the C: drive, it assumes that there is another operating system to preserve. So it copies the previous boot sector to a file named BOOTSECT.DOS before installing its own boot sector and a handful of hidden files.

The NT boot sector loads a hidden program named NTLDR (the NT Loader). That program displays a boot selection menu based on information in a plain text dataset named BOOT.INI. The contents of this file looks something like this:

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT Version 3.51"
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT Version 3.51 [VGA mode]" /BASEVIDEO
C:\ = "MS DOS"
```

The [boot loader] section give the user a 30 second time-out to select one of three options from the menu. If nothing is entered, then NT is loaded by default.

The remaining lines are in a cryptic notation that can be blamed on the ARC (RISC computer people).

Let enough technicians attend enough meeting and they will produce something like this. A computer can have more than one disk controller. Some controllers may be IDE and some may be SCSI. A controller can have more than one ribbon cable (or bus). A cable can have more than one disk. A disk can have lots of partitions. Adapters, cables, disks, partitions, how many were going to Saint Ives?

The ARC standard allows an operating system to be on any partition of any disk on any bus on any controller. The first word is "multi" for IDE (or for a SCSI disk controller masquerading as IDE to the BIOS), and "scsi" for a SCSI controller that cannot be accessed through the BIOS. The string "multi(0)disk(0)rdisk(0)partition(1)" is ARC notation for "the C: drive".

```
multi(0) is the first IDE disk adapter card
disk(0)rdisk(0) is the first IDE disk on that adapter card
partition(1) is the first Primary Partition on that disk
```

\WINDOWS is the directory that contains the NT operating system.

Das Boot

Howard Gilbert
(PC Lube and Tune Classic)

ARC notation gets more interesting when an Adaptec 3940 PCI SCSI-2 adapter with two buses is used to connect up to 14 devices to an NT Server. Then the various ARC elements can be used to select a particular adapter, bus, SCSI device ID, and partition.

Windows NT has a one megabyte driver that can be used on an Intel PC to load the operating system off any supported SCSI adapter. It has to be that large because there are lots of SCSI cards from lots of different vendors, and each is programmed differently. However, if the operating system is installed on one of the first two disks on the first SCSI adapter, and BIOS support is enabled on that adapter, then the adapter will emulate IDE disks. NT will define the adapter as a "multi" instead of a "scsi" and can save the disk space and loading time required for the one megabyte driver program.

ARC notation requires that Primary Partitions be counted first, then logical volumes in an extended partition. This produces a problem if NT is installed in a logical volume, and then some other operating system or utility creates a new primary partition on the same disk. Even if the new partition contains no recognizable NT file system, it will still cut in front of the logical volumes and bump the NT system to the next higher partition number.

To correct partition number problems, boot DOS and issue the following command:

```
C:\> attrib -r -s boot.ini
```

Then use any DOS editor to change the appropriate partition numbers. Save the file away and reverse the process with the same command but using the "+r +s" switches.

Windows NT and OS/2

It is fairly simple to run Plain Old Windows, Windows NT, and OS/2 all on the same system. Mostly, you need a reasonable sized disk. It is not, however, easy to put all three in a single boot menu, and this is certainly the case when they share a common C:\WINDOWS directory.

The cleanest solution is to accept a two-level menu sequence. Install OS/2 on D: or one of the higher disk letters using Boot Manager. Install NT on the C: drive along with DOS. Label the C: drive as "DOS+NT" to the Boot Manager, and mark the Boot Manager active/startable to the MBR. Now when the system boots, the first menu will offer a choice between DOS+NT and OS/2. If the DOS+NT option is selected, the NT menu takes over and offers a choice of either NT or DOS.

Nobody wants the two menu situation, but the need for it is driven by several gotchas:

- The OS/2 Boot Manager loads boot sectors from the first 512 bytes of a volume
- The OS/2 system is loaded by selecting the boot sector of the volume containing the \OS2 directory.
- The NTLDR program cannot load real boot sectors from a real disk volume. Instead, it either loads the kernel from X:\WINNT\SYSTEM32 or it loads C:\BOOTSECT.DOS.
- The NTLDR appears to have to reside either on C: or on a floppy disk in the A: drive.

If OS/2 were the only operating system on C: before the NT installation, then the OS/2 boot sector would have been copied to BOOTSECT.DOS. Despite the name, BOOTSECT.DOS contains the boot sector on the C drive before the NT installation begins and it can belong to any operating system.

Das Boot
Howard Gilbert
(PC Lube and Tune Classic)

Then the NT menu would be able to select between NT and OS/2. Real DOS would have to boot off a floppy.

In a similar manner, if NT is already installed on the C: drive, and OS/2 is then installed in Dual Boot mode rather than using the Boot Manager, then the OS/2 dual boot mechanism will swap between OS/2 and NT as easily as it swaps between OS/2 and DOS.

How to Recover Trashed NT Boot Files

At the end of the installation process, Windows NT creates an Emergency Repair diskette. If the boot files are clobbered, they can be reinstalled by booting off the original Installation diskettes, then indicating to the first panel that a Repair is needed. One of the repair options is to restore the boot files (boot sector, BOOT.INI, and NT* hidden files).

A full repair requires access to all the files used to install NT in the first place. The Repair Diskette will contain a list of the subset of NT files that were copied to the SYSTEM32 directory and its subdirectories. It will also contain the Registry and other configuration files built in response to questions posed during installation. However, the programs and libraries are still on the installation diskettes.

The full repair compares the copy of NT files on the hard disk with the original copy on the diskette or CD. If a needed file has been altered or deleted, it can be reinstalled from the original source. If an update had been applied to the system, it will be removed by doing a full repair.

Can NT Boot off a Floppy?

The \WINDOW\SYSTEM32 subdirectory cannot be stored on a floppy. However, it is possible to format a floppy using the Windows NT version of the FORMAT command (thereby writing an NT boot sector on the floppy). It is then possible to copy the hidden files BOOT.INI, NTLDR, and NTDETECT.COM onto the floppy. It is then possible to boot the floppy and get the same menu that one would get booting off the C: drive (to select between DOS and NT). Once a selection is made, the remaining programs must be loaded off disk. So NT can boot the menu program off the floppy, but it can't boot the operating system itself from the floppy.