Get the most out of your PC

Acrobat Reader: How to ...

F5/F6 open/closes bookmarks - **F4** open/closes thumbnails

In menu View you can set, how the file is displayed

CTRL+0 = Fit in Window, **CTRL+1** = Actual size, **CTRL+2** = Fit width

You can set **SINGLE PAGE**, **CONTINUOUS VIEW** or **CONTINUOUS FACING**

.. try them out and you will see the differences.

Navigation

ARROW LEFT/RIGHT: forward/backwards one page

ALT+ARROW LEFT/RIGHT: same as in a browser: forward/back

CTRL++ zooms in **AND CTRL +-** zooms out

Is this booklet for you?

This is *not* a normal beginners' book; it is more a supplement to the many beginners' books that are already available. The contents are of varying degrees of difficulty. You will get the most out of this book if you are running DOS 5 or above. It includes a separate section on DOS 6.

I do not deal with Windows 95 very much. However, a lot of things in this booklet are still relevant when you use Windows 95.

My purpose

One of the aims of this project is to publish and distribute KnowWare booklets in as many countries as possible, thereby helping as many people as possible to use their PCs.

If you would like to support the project, please tell others about the guide and/or make suggestions to improve the text.

Important

I have tried to keep the contents as up to date as possible. Remember that the first edition was written in the beginning of 1993. Even though everything happens very fast in the computer world, most of us continue using programs longer than their developers would wish. Therefore, things are not happening quite as fast as many would like us to believe. *We* decide how fast things develop, because we users pay the bills. Remember that!!

The KnowWare philosophy

Is to pass on relevant and easy-to-understand information for a reasonable price.

My main motive is to pass on what I know about PCs to as many people as possible – as cheaply as possible. Computer books are generally too expensive (or rather, were. The publication of KnowWare books has forced prices down to a more reasonable level in Denmark). Users want information and knowledge that makes their lives easier. Whether this comes from a fine book with a four-color cover or from a booklet like this is unimportant. How much it costs *is* important.

The booklets are printed on a *rotary press* of the type used for printing newspapers. They are printed on long rolls of paper, all pages and on both sides simultaneously. Ten thousand copies of a booklet can be printed in a couple of hours, after which a bookbinder trims them down, staples them and packs them. I really like how they can be folded right back without damaging them. They can be folded and laid down next to your computer, or put in your back pocket.

About myself

I was born 1952, live in Denmark, and speak Danish, English and German. Between 1970 and 1980, I studied sociology and psychology at the University of Copenhagen, and also spent some years in Germany. I have worked with PCs since 1986. From 1988 to 1991, I worked with a main-frame computer.

Thank you

To everyone who has supported me and taught me many different things. Also thanks to all of you who have helped to improve and publicize this booklet. Heartfelt and grateful thanks to my other guides.

Happy reading! ☺

I hope that this booklet will bring you a greater understanding of your PC, insight into its secrets and more pleasure when you work with it.

☞☞ Important ☜☜

Something I must point out: everything you try using ideas or suggestions given in this text is *your own responsibility*. That's my disclaimer to avoid any litigation!!

My primary experience with PCs comes from so-called "clones," i.e. imitations of the IBM PC.

If you are impatient and want to edit your CONFIG.SYS right away, then *please* remember to have a boot diskette that *works*. Put a formatted disk in the disk drive and type

```
C:\>SYS A:
```

and also read *Boot diskettes*, on p. 55.

My advice is to read all text in the order it appears, including the "references" (which admittedly force you to jump around a little).

This applies especially to CONFIG.SYS if you have DOS 5, where you can easily write something that makes you unable to start your computer from the hard disk, i.e. it locks your PC.
Now you have been warned. This problem does not exist with DOS 6.

If you are experienced at editing startup files, you may begin with *CPU and memory* on p. 12. This chapter contains something new and relevant for most readers.

If you don't understand much in these chapters, just make the suggested changes in your startup files – or have someone else do it for you. As we go on, I'll try to point out what is technical, what is advanced and so on. I assume that you have installed DOS in the directory C:\DOS. References to Windows refer to version 3.1.

Jargon

The PC world is filled with jargon. I will try and explain some of these expressions.

DOS commands and lines in files are written `like this with Courier`.

If I write "write in DOS" or "type,", it means do so at the DOS prompt (`C:\>`), i.e. you are "in DOS" and can enter DOS commands. Some programs can temporarily jump to DOS and then return when you type EXIT.

When you are "in" a directory, (you should be in C:\EXTRA) it means that the directory is active and on your screen, i.e. the DOS prompt in this case appears as `C:\>EXTRA` and DOS commands (without further specification) will be carried out on files in this directory.

Default is a very good concept or word, although difficult to understand if you haven't met it before. In a computer context, it refers to *that which is chosen automatically unless something else is specified.* A couple of examples: if you are in a directory and type `DIR`, DOS says: "As you are not telling me which directory you want details of, I'll choose the default for you" – and default here is the actual directory you are in. When you are at the DOS level, you are always in a *directory* and on a *drive*. I'll explain *directory* later.

Within programs, you will come across *default settings*. When you start with an empty document, such things as left and right margins, choice of font, line spacing, and so on are set at *default values*. When you want to load or save a document in your word-processor, a *default directory* is used unless changed. Usually, the user can alter most or all of these *defaults*. I hope that you have an idea now about the meaning of *default*.

The *root* means the root directory, the "first" directory on a disk. In this text, I shall be referring frequently to the root directory on the hard disk, which is C:\

File names are usually written like this: HIMEM.SYS (small caps).

Directory names are written like this: C:\DOS (ordinary caps).

(A) begins a paragraph, or several, of what I would consider *advanced* text.

All references to the "manual" mean the Microsoft MS-DOS 5.0 manual. DOS 6 is treated in its own section.

Please read this

First, a word of comfort: this chapter does *not* contain any technical material. *So please read it!*

Over the past few years, I have helped many friends and have seen how much time one spends learning to use the PC and its programs. If you run into difficulties, it *can* take a really long time to solve them – and not everybody wants, or can afford, to pay others to help with a problem. I hope to be able to reduce this time for you, *but you have to be willing to invest some hours* in the first place. It will pay you dividends in the long run.

If you are a beginner, don't be nervous. Millions have learned it before you, but it takes time. In the beginning, you might only get something from a few chapters but later on you can investigate the others. Not so many years ago I didn't understand a single word of what I write about today. As we all know, the best way to learn is to teach others.

A comforting thought: the later you've started in the PC jungle, the faster you'll be able to understand and use your PC. Programs and their on-line help are getting better, and there are more and more books.

There are certain basic things that must function before your PC works well. In the long run – if you don't have it already – you'll probably have to adopt a sense of order and discipline if you really want to gain a lot from your computer.

It is a sad paradox of this computer age that books, magazines and people's brains contain a wealth of information – but nobody knows exactly where it is or how to find it quickly. The enormous amount of *information* as opposed to relevant, useful *knowledge* is really a problem. Nobody can know everything in this business because it all happens so fast, the market is very big and new products are arriving in an ever-rising flood.

One reason this guide has become necessary is the inferior manual supplied by Microsoft. There isn't a single example of the all-important startup files AUTOEXEC.BAT and CONFIG.SYS in 600 pages.

The more I wrote about the technical aspects of my subject, the more the text dwelled on the theme of optimization, i.e. tuning, trimming, getting the most out of the computer. Some sections of this booklet have become more advanced than I originally intended but those who understand how to use the advice will be pleased with it and, I hope, will help their friends a little.

The start files are not the most exciting part of a PC, but they areone of the most important things to know about, which most computer developers conveniently forget.

One of my aims when I started to write in September 1992 was to write *the book I wanted*, the book I wished I'd had when I had problems on my PC.

Because the text is of varying difficulty, it has not lent itself to a presentation in easily graded sections, designed for reading through once only. You may miss a "yellow brick road" to follow. If you are an experienced PC user, you may sometimes find yourself in the middle of a beginner's course.

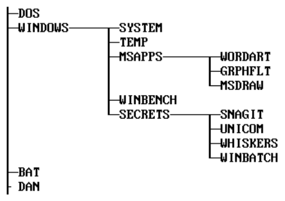
Some readers may find that they have to browse through to orientate themselves, and then start again at the beginning. For those of you who want to use the book as a reference in the future, there is an index at the back. A *term* is defined, as far as possible, the first time it occurs.

I have written on other subjects before but I have to admit that it has been hard to structure a PC book in which the subject can alternate between simple and very difficult. It isn't always easy to find the right balance between theory and practice. At least I have one advantage over other authors: you can't criticize me about the price!

I assume that you are ready to do some donkey work – in other words, *you want to think for yourself!* You can find – and solve – thousands of specific problems on a PC. I have tried to pass on *theoretical knowledge* illustrated with *relevant specific examples* that I hope will also give you enough knowledge to be able to solve problems *other* than those described in this text.

Directories and files

The concepts of *directories* and *files* are so intertwined that you may need to read about one before you understand the other. I have chosen to start with directories. I am assuming, for the sake of simplification, that you have only two disk drives: `A:` (the diskette drive) and `C:` (the hard disk) and that you have no other *logical drives* on your hard disk. Logical drives are explained below.



Later, I shall be giving you a short description of how a hard disk is built up. In the meantime: *Direct*ory means signpost, that which *directs*. Just as most of us live at an address, so that the postman and others can find us, we can say that a file also has an address on the hard disk – or on other media like a floppy disk. The “needle” (*read head*) of the hard disk has to find the file, especially the physical place where the file starts.

Imagine there is a wide highway, crossing the whole country. This corresponds to what is called the “first” or *root* directory. Every side road off that main highway corresponds to a sub-director. Every side road off each individual side road corresponds to sub-directories to the first sub-directory, and so on. We live along these roads, and we all live at a house number.

The left of the diagram shows the root and some of my sub-directories. I have five sub-directories under Windows, some of which have no sub-directories.

Let us choose the file C:\CONFIG.SYS. The file’s address is specified by a name, which is made up of the following components:

`C:` designates the *logical drive*; the colon helps to identify it as something special and not the name, for example, of a file. DOS uses the concept of logical drives. Other logical drive designations D, E, F, and so on correspond to other “countries.” The historical reason is as follows: the first PC design was diskless, the next had one disk drive, the next had two, the next had a hard disk fitted and so on. A: is the first floppy disk drive, B: the second, C: the first logical drive on the first hard disk, D: is either the next on the first hard disk or the first logical drive on the second hard disk and so on.

`\` is called a *backslash* and denotes the start of a (sub-) directory.

`CONFIG` is the file’s *name*.

`SYS` is the file’s *extension*.

When you first got your PC, you – or the programs you installed (e.g. WordPerfect) – began by creating new directories. For instance, C:\DOS and maybe C:\WP or C:\WP51. These are sub-directories (side roads) off the root directory. They are where DOS program files and WP program files live. WordPerfect’s chief file, the main program, is – to give it its full name and address – C:\WP51\WP.EXE.

When you type WP, or start the program any other way, this file is activated. Again, the full name is a drive (a letter followed by a colon) plus a sub-directory plus a file name and extension.

Notice the rather irritating backslash **`\`**, which is used as, and called, a separator. The ASCII value is 92.

Why is it that you should divide your hard disk into all these directories? To make it easier to find and manipulate different programs and files. It is very practical to have all the files that belong to WP in one directory, perhaps with some attached sub-directories. There they all are if later you want to move, copy or delete the whole program.

When you want to start a program, you have to point DOS in the right direction by stating which directory and which file. At the prompt, you can move to the relevant directory by using one or more *change directory (cd)* commands, e.g.

```
C:\>cd wp51
```

and then start WP by entering

```
C:\WP51>wp
```

WP starts because the file WP.EXE is in the default directory, which here is `C:\WP51`.

Now if you want DOS to look in directories other than the default directory when you start a program, you can define a special *path* (collection of directory names) that DOS will remember. If in any directory you type

```
C:\>path c:\dos;c:\wp51
```

then on the command

```
C:\>wp
```

DOS will look first in the root directory (`C:`) for WP.COM, then for WP.EXE, then for WP.BAT, and finding none of them here it will look in the first section of the path (`C:\DOS`), then the next

(`C:\WP51`). A semicolon separates the different directories.

If you have problems starting a program, e.g. you receive the message `Bad command or file name`, the cause could be that your *path* doesn't contain the directory in which you have the program. Check your AUTOEXEC.BAT – it normally specifies the `path` command.

When you save your first document, you should notice what the default directory is. In other words, where is the computer placing your documents? If you don't keep an eye on this you'll find it hard – at least in the beginning – to find them again.

A directory is a storage area containing files. Every disk (whether diskette or hard disk) has what is called a *FAT* (**F**ile **A**llocation **T**able). It keeps track of the physical location of directories and files on the disk and is obviously *an extremely important element* because without it, DOS cannot keep track of data.

Every time you create, save, copy, delete or do anything with one or more files, the FAT is updated. Therefore if you delete 50 files in a directory it takes some time to update this table, which is essential on a PC. In fact there are two FATs and if one of them becomes unreadable, DOS tries to repair it by referring to the other. Luckily this doesn't happen often but it can. This is one reason that it is important to make back-ups.

When we work with files using a program that is written specially to manipulate files (like Dosshell, Norton Commander, PCTools or Windows File Manager), we usually see a stylized graphic picture of the *logical* structure: the arrangement of directories and files on the hard disk.

The "main highway," the root, comes first with all its sub-directories. As we have seen, every one of these sub-directories can have its own sub-directories, and usually there is one file or more in every directory. Nearly all programs show this logical arrangement – actually a picture of the FAT – in a form known as a directory *tree*.

The root directory (`C:\`) should contain *as few files as possible* in order to keep your PC lean, mean and fast. Preferably only the two startup files plus COMMAND.COM – here I am not considering the "hidden files" that are part of the operating system. Unfortunately certain programs place one of more of their files in the root directory.

If on inspection you find you have a lot of files, find out if you really need them in `C:\` and, if not, delete them or move them to the relevant directory.

Files

It is of fundamental importance that you manage your files well, store them in obvious places and know where to find them. You can do much of this at DOS level or in some applications, e.g. WordPerfect, but personally I use and prefer the file management program Norton Commander, which I discuss on page 33.

In a computer context, the word *file* refers to a specific collection of data. The data may make up part of a computer program (or a whole program) or a company report or a letter to your grandmother or the latest edition of your school newsletter. In the old days (and to some extent it is still the practice), letters and reports were stored in hanging folders in a filing cabinet. You can think of each hanging folder as a *directory* that contains *files*.

DOS, the operating system, consists of many different files, each of which performs its own special purpose so that other programs, for instance a word processor, can work. If you write `DIR` at any DOS prompt, the screen will show you the names of all the files in the directory plus their sizes, creation dates and so on.

Your word-processing program also consists of a collection of files, each of which executes a function. When you install modern programmes they often ask you in which directory you will place the program's different files, and they usually make their own suggestions, e.g. C:\WP51 for WordPerfect 5.1. I've chosen WP as an example because it is so widely used.

Different types of files

There are two basic types of files: those that form part of a program, and those that contain data that you and your program created. WP.EXE is part of WordPerfect but a letter to the tax authorities is called a document or data.

If you create a directory listing (`DIR`) or look at the file names using Dosshell, Norton Commander or a similar program, you will see that they have different extensions.

An extension of COM or EXE indicates a program file. COM files can have a maximum size of 64 KB, while an EXE file can be larger.

CONFIG.SYS contains *calls* for many different device drivers, which usually, but unfortunately not always, have the extension SYS. Other "driver files" can carry the extension DRV, typically in Windows. *Batch files*, which I will describe later, *must* have the "surname/last name" You can start

BAT. COM, EXE and BAT files simply by typing the name of the program (file) at the prompt.

Data files (letters, reports, shopping lists, and so on) are nearly always given an extension automatically by the program that produces them. In the following paragraphs, I will discuss what are known as straightforward *text files*, i.e. files that contain *only* text with no formatting codes (bold, underline, etc.).

Documents from word processors are often given the extension DOC by default. Traditionally, text files are given the extension TXT or ASC. A text file has been stripped of all the various codes that a word processor normally includes to show where the margins are, whether letters are bold, in columns and so on. Other names are *ASCII files* or WordPerfect's peculiar expression *DOS files.*

You can use an *editor* (a mini word processor) to write or change a text file. Microsoft now includes an adequate one called EDIT with MS-DOS.

The good thing about standardized extensions is that you can quickly see what kind of file it is. Fortunately, everyone seems to agree on the same standard. More and more programs have text files included with them that are read by the main program when it starts up, and are used to configure the program or set up default values and other settings, so that the program runs in a particular way.

Windows with its text files WIN.INI, SYSTEM.INI and SETUP.INF is the best known, and most difficult, program in this regard. But generally more and more programs have an INI (initialize, beginning or startup) file – which is read when the program is loaded.

CONFIG.SYS and AUTOEXEC.BAT are in effect DOS's initializing files. Goodness knows how many millions of people have spent how many millions of hours over the years typing these strange file names!

If I am laboring this point somewhat, it is because PCs and their programs are becoming increasingly complex. Text files containing important information about the way a program works are often altered by other programs. It is important to be able to look at and amend these files.

Edit and BAT files

If you aren't used to editing text files or creating batch files, you are now about to participate in a mini-course. But first a warning! *Don't begin to alter your **startup** files until you know what you are doing!* The responsibility rests with you.

EDIT.COM starts a text editor called QBASIC.EXE, which we will use to create and alter a *batch file*. It can also create programs in the BASIC language.

At the DOS prompt, enter `EDIT` to see if the program starts. If not, perhaps C:\DOS isn't in your PATH in AUTOEXEC.BAT, or perhaps one of the files is missing. If it starts, press Esc, Alt+F and X to close it.

I suggest that you have a directory called BAT where you can store your batch files. If you do not already have one, you can create one by typing

```
C:\>MD BAT
```

Move to this directory by entering

```
C:\>CD BAT
```

What is a batch file? Well, if you type the same DOS commands time and again, it can pay to write these commands in what is called a batch file, a text file containing DOS commands that has BAT as its extension.

The following is a quick way to create a text file. Try it, just to see the principle behind it. Let's make a file called EASY.BAT. (note: `copy con` means copy from the console, the console being a term covering both the screen and the keyboard).

```
C:\BAT>copy con easy.bat
```

The cursor blinks and you are "in" a text file. Type

```
c:
cd\
dir
cd \dos
dir/w
cd \bat
```

Press F6 to end the file, then Enter. DOS displays `1 file (s) copied.` `Copy con easy.bat` means "copy from the console (keyboard)," i.e. copy what I type into a file called EASY.BAT. The file EASY.BAT has been created. Now, just by entering EASY you can "run" this batch file. Try it. It shows the contents of two directories. By pressing Ctrl+S, you can stop and start the process if you want to follow it. (Ctrl+S means to hold the Ctrl key down while you press the S key.)

Now make a batch file called ED.BAT, which will automatically start EDIT and load EASY.BAT in the editor.

```
C:\BAT>copy con ed.bat
```

```
edit C:\bat\easy.bat
```

press F6 and Enter.
When you now enter ED and Enter, EDIT will start and load the file C:\BAT\EASY.BAT.

The EDIT editor consists of a rectangular area where you can type and edit as in a simplified word processor. The name of the file you are working on is shown at the top of the screen. At the top and bottom, there are horizontal bars. The top bar contains the menus used for editing.

All of the words FILE EDIT HELP are "headlines" for independent menus that drop down from the bars like a roller blind when you click on the word with your mouse. If you are using the keyboard, then the menu bar is activated by holding the Alt key down while you press the underlined letter of the menu command you want, e.g. open the File command by pressing Alt+F.

This is how you move a line – which can be relevant if you want to edit a startup file using this editor. Place the cursor at the beginning of the line. Hold down the Shift key and press the "down arrow" key. The line is marked (highlighted). Press Shift+Del(ete). The marked line vanishes from the screen and is placed in a temporary store. Move the cursor to the point where you would like the line to appear and press Shift+Ins(ert) and the line is inserted.

Close EDIT with Alt+F, X. If the file has changed (which in this case it has), you will be asked if you want to save it. The Tab key moves you between the possibilities or you can just type the initial letter. In this case, answer NO to keep your original file. Now create the following two batch files. EC.BAT starts the editing with CONFIG.SYS loaded, and EA.BAT starts the editor with AUTOEXEC.BAT loaded. The two directories DOS and BAT should be in your PATH.

```
C:\BAT>copy con ec.bat
edit c:\config.sys
```

Press F6 and Enter.

```
C:\BAT>copy con ea.bat
edit c:\autoexec.bat
```

Press F6 and Enter.

But remember those early words of caution:
don't begin editing your startup files until you know what you are doing!

You may notice with EDIT that there is no "word wrap" – where the screen shows a new line even though you haven't put in a carriage return.

Text editors work in different ways but most of them work like this: text between two carriage returns (a carriage return is when you press Enter) is shown as one line that continues past the monitor's right-hand boundary, and can contain up to a maximum of 256 characters. You will have to get used to this. But many key combinations, such as Ctrl+arrow, work exactly the same as in other programs. It is an advantage that a text editor doesn't have word wrap: the computer reads BAT and INI files *line by line*, so it could cause havoc if a line gets broken unexpectedly into two lines – as if it were two sets of instructions.

ASCII

This is only relevant for DOS, not Windows. Understanding ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) was one of my first breakthroughs many years ago. It is pronounced “aski” and is a standard for the relationship between a symbol and a number.

When you press a key on your keyboard and the screen shows a character, it is nice to think that the character is just sitting in the key, just waiting to be activated. But it is not so simple.

I am going to jump over the *scancode*, which is what is actually sent when you press a key. This code is sent to a “converter” that decides which ASCII value is linked to the scancode, and then this ASCII value is again “converted” to a character on your screen.

In computing, only two states are possible: on or off, 0 or 1, *and nothing else*; and this fills 1 bit. A computer can only move 1s and 0s around, nothing else! Eight of these 1s or 0s grouped together are called a *byte*, and the letters and symbols you use are each defined by a series of 1s and 0s assembled into an eight-partition electronic box. Each partition can contain either a one or a zero (binary system). With eight partitions, each of which contains one of two (0 or 1) available values, there are 256 possible patterns (2 multiplied by itself eight times) for creating a number value. As zero is also a valid number, you will often hear of the set of values referred to as 0-255.

Fortunately, computer manufacturers are pretty much in agreement over which symbol or letter should be allocated to which of the 256 values in the ASCII system. Let’s use the capital letter “A” as an example. At the DOS prompt, or in a DOS word processor, try holding the left-hand Alt key down while you key in two or three digits on the numeric keyboard, for instance 65.

Then, to make everything a bit more fun (though easier for programmers) when they created Windows, Microsoft introduced another standard called ANSI, which uses other character values.

Luckily, the normal letters and numbers are unaltered, but a text file stored as ASCII and one stored as ANSI are two different things. Text files saved by DOS programs are in ASCII format, while text files saved by Windows are in ANSI (by default, anyway). Write and other Windows word processors are able to save in either format and convert between the two.

These are some of the foundations of *data communication*. We humans have not really got any further in mimicking the brain and soul’s development, but it is progressing. The only reason for the computer’s wide use is its speed. Forget about “intelligent” computers, even though experts are trying to convince us it is possible with talk of neural networks and the like.

16-number system

(A) Number values can also be represented by other number systems. In the computer world, numbers to base 16 are used a lot. The system many of us had banged our heads into at school is the normal decimal system to base 10.
0123456789 and then (because we haven’t any more symbols [fingers or toes]) we set 0 in the first place, to the right, and put a 1 in the second place, to the left, and we have 10.
Hexadecimal, the base 16 system, looks like this: 0123456789ABCDEF and only then – with the value we write as 16 – do we come to 10.
In the base 10 system, we say: the last cipher is the 1s, the next is the 10s, the next is the 100s and so on. In base 16 (because we are still thinking in terms of base 10), we say the units go as high as F (=15), the “10s” are the 16s, the “100s” are the 256s and so on. Now that’s not so difficult, is it?
The decimal system is inappropriate for computer operations. It is a relic from man’s early days, when we thought using 10 fingers.

binary	10	16
00010000	16	10
00100010	34	22
11111111	255	FF
	256	100
	65536	10000
	1048576	100000

The computing world handles large values and the hexadecimal system has proved its worth. Numbers are shown with a final “h,” e.g. A0000h. The last zero is often dropped, so the above becomes A000h, which in the decimal system is the familiar 640 KB = 640 x 1024 bytes = 655360 bytes.
With the Windows calculator (choose View, Scientific) you can amuse yourself with these numbers and convert them easily.

Memory

Some of the following is rather technical and can be skipped. Memory is where the PC stores information. It uses two types of memory: RAM and ROM.

ROM (Read Only Memory) is not discussed in great detail in this guide. ROM is a fixed form of memory (built into electronic chips) that “remembers" certain parts of the operating system. Other ROM is in physical parts of the PC like the graphics adapter and the hard-disk controller.

RAM (Random Access Memory) is the volatile memory that’s available for calculations while the PC is switched on. The CPU (Central Processing Unit) uses and processes data in the RAM, where the contents are constantly changing.

First, something about the allocation of RAM in a PC, a complex subject, if the operating system is DOS. Computers, as we have noted, use the binary system. You will often come across numbers raised to the power of two: 2, 4, 8, 16, 32, 64, 128, 256, 512 bytes and so on. When you get to 1024 bytes, it is called 1 kilobyte (KB or K); thereafter you follow the same pattern of 2, 4 ... up to 1024 KB, which is equal to one MegaByte (MB), and so on until you reach 1024 MB = 1 GigaByte (GB).

When IBM built the first PC, they did not think that it would become the success it is and so they figured that 1 MB (1024 KB), with 640 KB set aside for programs, would be adequate memory for all purposes.

Let’s look at the first MB of RAM, the first 1024 KB. Imagine a giant bookshelf with lots of spaces for identical books. Each space is the space for one byte and has what is called an address. What follows describes how these areas are defined and treated.

The work of the CPU consists mainly of moving all these bytes around between the hard disk, itself, RAM, the monitor, etc. – fast. Virtually all communication in a computer – the transport of data between the different units (devices) – has to go via the CPU. Monitor, keyboard, disk, and so on are physical devices, but other “logical” or abstract parts of a PC can also be “devices.” As we shall see later, all devices must be controlled, or handled, by a special program called a *device driver* (or simply *driver*). It took me years to understand the strange concept of a *device*!

DOS, which was originally created for the old 8086 processor, can only “see” the first MB. When the processor runs in its most primitive mode, *real mode*, it can only “see” this area in RAM. DOS is a *real-mode* program. When you boot your PC, the processor “wakes up” in this real mode and looks around for its partner, DOS. That’s how it has been ever since DOS arrived on the scene.

If you want more from your PC than real mode and DOS, which can only use 1 MB, you have to build on these foundations.

So as not to exclude the 8086 and 80286 and other early PC versions, DOS 5 and 6 are still real-mode programs and are “backward compatible,” which means that all programs written for earlier versions still work with newer versions.

To use RAM above 1 MB, you need a program that makes the processor run in what is called protected mode. *Protected* means stopping two programs that are using RAM simultaneously from trying to use the same areas of RAM. The 80286 was the first processor that was able to run in protected mode as well as in real mode.

The processor can only be in one mode at a time and it takes time to change between real and protected mode.

A CPU works at a certain speed, called its clock frequency, which is the number of pulses per second that the current flows (Hertz). This is typically a figure like 33, 40, 75, 90, 133 or higher, measured in MHz (millions of Hertz – not a misprint, it really is millions of times a second!!).

Forms of memory

This section is rather technical, but helps you to understand what is going on before we get to the start files.

1024-	extended high=first 64 K
640-1024	upper
0-640	conventional

Conventional memory

The area in memory from 0-640 KB is called *conventional memory* or lower memory. This is where most of the work is done. DOS is the memory manager for conventional memory (it controls it).

Upper memory area

This section is included because it is important, when you use DOS 5 or higher. From DOS 5 onwards, it is possible to load certain programs into upper memory – which is an address area. This frees more space in conventional memory to run DOS programs.

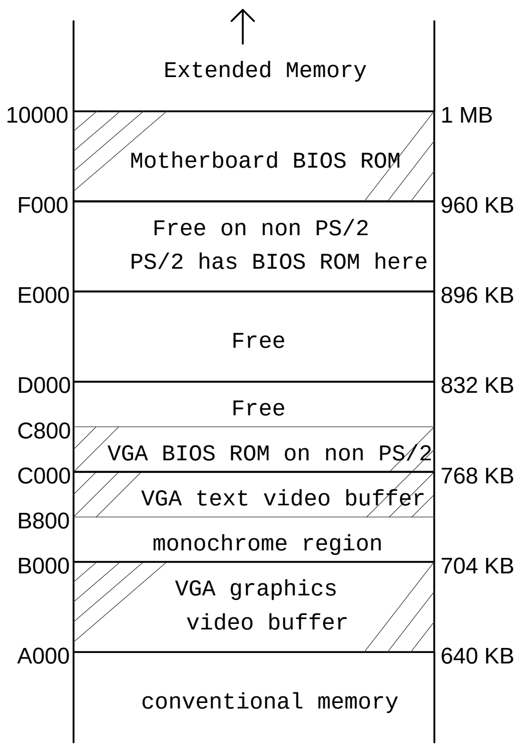
Don't worry if you don't understand a thing. I have not understood all the details myself yet but each time I write about it I understand a little more – I hope – and I have worked intensively with this for many months.

Sometimes I wonder how on earth sensible people could have constructed a PC with an operating system, memory, and so on as difficult as it is – but they have. Part of the problem is that, regrettably, IBM couldn't foresee the future many years ago!

IBM wisely reserved the address area from 640-1024 (384 KB) for system, technical and future purposes. This reserved address area in memory from 640-1024 KB is called, by most people, *upper memory*. IBM and Hewlett-Packard sometimes refer to it as *reserved memory* – when the terms *high* and *upper* memory were introduced, the industry was in some confusion!

Upper memory area is the name for an *address area*. There are rules as to where in upper memory the different *devices* must place their data so that the CPU can have direct access to communicate with them – the marked areas on the chart show where.

But the whole area is not occupied completely; the vacant space left amongst these addresses can



be used for drivers and resident (TSR) programs. *Resident* means that the program is permanently in RAM and there is no need to call it from the hard disk every time it is used. Resident programs are also called TSR (**T**erminate and **S**tay **R**esident).

The interesting thing is that the area lies within the reach of DOS and real mode. DOS can see and control/communicate with them, so DOS drivers and DOS resident programs can be run in this *address area*.

In recent years, methods have been developed to allow DOS to place some of its files in this address area. DOS does this using EMM386.EXE, which can provide this access – this text does not deal with programs such as QEMM, 386MAX and Netroom.

The term *upper memory area* is not the same as **U**pper **M**emory **B**locks (UMB), although these blocks – if created, for example, by EMM386.EXE – are from the memory point of view situated in upper memory's address area. Upper memory is a name for an address area. UMB is real useable RAM in this area.

The memory manager for upper memory can be either HIMEM.SYS or DOS. In the early stages of the booting procedure, HIMEM.SYS is the memory manager. The usual thing though is for DOS to take over as memory manager via the command `DOS=UMB` inserted in CONFIG.SYS.

The illustration is an example of how upper memory can look on a PC, and can maybe help you if you begin to investigate where in this area different programs load themselves.

IBM PS/2 puts BIOS ROM in the two “top” areas, i.e. E and F page. VGA BIOS ROM is placed from E000-E7FF. This leaves the area C000-E000 free for UMBs.

Expanded and extended memory

It is important to distinguish between the physical RAM or memory and the abstract names (upper, extended, etc.) for the way physical RAM is used. If, for instance, you have 4 MB of physical RAM, you can use this in many different ways. This section deals with this. To specify the way in which it is used, you speak of configuring memory.

In the “old days,” the use of memory above 1 MB started primarily with the Lotus spreadsheet *Lotus 1-2-3*, with users soon hit the ceiling of the 640 KB, which was the limit for DOS programs. Lotus, Intel and Microsoft developed rules for how their version of *expanded memory* should be used. This standard was called **E**xpanded **M**emory **S**pecification (EMS) or LIM EMS.

At this time, extra memory cards were manufactured to put in the PC. Primarily older programs are able to use this form of memory, though even many newer games need EMS memory (see Multiple Boots, page 47) A memory manager for expanded memory is called an **E**xpanded **M**emory **M**anager (EMM). DOS has one of these called EMM386.EXE.

Later, another standard for using memory over 1 MB was introduced: e**X**tended **M**emory **S**pecifications (XMS). It is most unfortunate that the names are almost identical. This is more relevant to most users today as newer programs are primarily written to use extended memory, which also requires a memory manager, a so-called XMM. DOS has one, HIMEM.SYS.

These terms are the names given to the use of physical RAM. They are standardized rules. So RAM above 1 MB can be used as a combination of different forms of memory – but not until the relevant memory manager is active and able to control how much to use and for what. Expanded memory is also called EMS memory, and when it is available, it is configured according to published standards.

High memory area (HMA)

The *High Memory Area* (HMA) is defined as the first 64 KB of extended memory, from 1024-1088 KB. Using a little trick, this area has become available as if it belonged to the first MB, so DOS can use it. Most people use this area to place parts of DOS using `DOS=HIGH`. Naturally, the memory manager for this area is HIMEM.SYS, as it handles all memory above 1 MB. So HIMEM.SYS controls both high memory (area) and extended memory.

`MEM/A` is the only DOS 6.x command that tells you how this area is being used. The switch is undocumented.

(A) HMA is apparently “a part” of extended memory – yes and no: HMA is available to the processor in real mode while extended memory is only available to the processor in protected mode.

How upper and extended memory work together

(A)Upper Memory borrows the actual RAM from extended memory.

In the jargon, extended memory is *mapped* to upper memory’s address area, i.e. the CPU is led to believe that certain addresses are in a certain place, though physically they are somewhere else.

For example, if you want access to, say, 92 KB of upper memory, it is borrowed from extended memory, which therefore is 92 KB smaller. Try writing REM at the beginning of the line in CONFIG.SYS containing EMM386 and see the result of `MEM`. Delete the REM again, boot up and check `MEM`. The sum of upper memory + extended memory is the same!

Devices

All these forms of memory, except conventional memory, are “a device" in the same way as a monitor, keyboard, printer, etc., and so these forms of memory require a device driver in the same way that other devices do.

There are both purely physical and what I call logical, or abstract, devices. The monitor and the keyboard are in themselves not devices but the logical term *con* or *console* is a device that refers to the two physical objects, an input and an output object.

The term *console* comes from the days of the original mainframe computers. These had no monitor or separate keyboards but received input and wrote out their results through a kind of telex machine. This machine, called the console, was used for all input to and output from the computer.

The startup files

The startup files CONFIG.SYS and AUTOEXEC.BAT configure the PC in a certain way, depending on their contents. DOS 5 and 6 are basically identical as far as these files are concerned. I have chosen to concentrate on DOS 5 with some references to DOS 6, which has a section of its own at the end. You should only consider these examples as suggestions. These files *must* be in the root directory on the boot drive, typically `C:\`.

When you install programs, they often suggest making changes to the startup files, so it is nice to know something about what the different lines in these files mean.

CONFIG.SYS will typically contain *calls* for device drivers, programs that handle devices. AUTOEXEC.BAT contains DOS commands that are automatically run every time the computer is booted. They *are* DOS commands, though most of them are a little special, so they could just as well have been run from the DOS prompt.

In both startup files, if `REM` or `rem` followed by a space is at the beginning of a line, the line is considered a remark and not a command and is therefore not executed.

There is no difference between something entered in capital or lowercase letters. Remember to have a working boot diskette handy before beginning to make changes in CONFIG.SYS. I have chosen examples with two codepages but the commentary describes what to do if you only want one codepage. *Codepage* is a strange term that describes a collection (a set) of 256 different characters that you see on the screen. Each codepage has a number.

The important files

HIMEM, EMM386, SMARTDRV, RAMDRIVE and MEM are *vital* files (given here without their extensions) for the operation of DOS 5, Win 3.1, DOS 6 and Win 3.11. The release dates of these versions of DOS and Windows are given deliberately, and the text deals with these files in *chronological* order. SMARTDRV is an exception. This disk cache program has been much modified and so it has its own special section.

DOS 5
HIMEM.SYS, EMM386.EXE, SMARTDRV.SYS and MEM.EXE. All these files are usually placed in C:\DOS. It is important here that SMARTDRV.SYS only functions as a device driver and can only be installed from CONFIG.SYS. This was SMARTDRV version 3.

Win 3.1
Included an improved version (4.0) of SMARTDRV in the form of an executable EXE file (SMARTDRV.EXE). It could now be run from AUTOEXEC.BAT. Win 3.1 also introduced a fine program, MSD.EXE, which can look at the first MB of memory. All these files are usually placed in C:\WINDOWS.

DOS 6.0
Had the same vital files as Win 3.1, but EMM386.EXE and SMARTDRV.EXE were significantly improved. All files are, once again, placed in C:\DOS.

So watch out if you install Win 3.1 *after* installing DOS 6 and at the same time accept the Windows 3.1 Setup option to change your AUTOEXEC.BAT automatically. If you do this, you won't be using the new versions from DOS 6. In this case, you should alter your startup files so that EMM386.EXE, and so on are read from C:\DOS and not from C:\WINDOWS.

DOS 6.2
The only change was SMARTDRV.EXE ver. 5.0.

Win 3.11
(Windows for Workgroups) arrived with the same files as DOS 6.2.

Specify the latest versions of these important files in your startup files. When you have done that, delete the old ones. This not only saves disk space but also avoids confusion. In the examples, I have used "`..`" (two dots) for the relevant directory and not "`[path]`," as is normally written.

I recommend that you read right through the relevant commentary on CONFIG.SYS *before* you begin to change things.

DOS 5 – SMARTDRV.SYS

CONFIG.SYS

```
device=\DOS\himem.sys
dos=high
device=\DOS\emm386.exe noems
dos=umb
devicehigh=\DOS\smartdrv.sys 512 256
devicehigh=\DOS\display.sys con=(,,1)
rem devicehigh=\UTI\gmouse.sys
devicehigh=\DOS\ansi.sys
rem devicehigh=\DOS\setver.exe
country=044,437,\DOS\country.sys
files=30
rem fcbs=1
buffers=5
rem stacks=9,256
break=on
rem shell=\DOS\command.com \DOS /p /f /e:1024
```

AUTOEXEC.BAT

```
@echo off
path C:\BAT;C:\DOS;....
mode con cp prep=((437) \DOS\ega.cpi)
mode con cp sel=437
LH keyb.com uk,,\DOS\keyboard.sys
rem LH \DOS\doskey.com
LH \DOS\share.exe
rem LH \UTI\gmouse.com
set temp=C:\temp
set tmp=C:\temp
prompt $p$g
rem \DOS\emm386.exe auto
```

Windows 3.1 or DOS 6 – SMARTDRV.EXE

CONFIG.SYS

```
device=\..\himem.sys
dos=high
device=\..\emm386.exe noems
dos=umb
rem devicehigh=\..\cdmke.sys /d:mscd01
rem devicehigh=\..\ramdrive.sys 2048 /e
devicehigh=\DOS\display.sys con=(,,1)
rem devicehigh=\UTI\gmouse.sys
devicehigh=\DOS\ansi.sys
devicehigh=\DOS\setver.exe
rem device=\..\smartdrv.exe /double_buffer
country=044,437,\DOS\country.sys
files=40
rem fcbs=1
buffers=5
rem stacks=9,256
break=on
rem shell=\DOS\command.com \DOS /p /f /e:1024
```

AUTOEXEC.BAT

```
@echo off
path C:\BAT;C:\DOS;C:\WINDOWS;C:\UTI;....
rem LH MSCDEX /D:MSCD01 /M:20
\..\smartdrv.exe
mode con cp prep=((437) \DOS\ega.cpi)
mode con cp sel=437
LH \DOS\keyb.com uk,,\DOS\keyboard.sys
rem LH \DOS\doskey.com
rem LH \UTI\gmouse.com
LH \DOS\share.exe
set temp=C:\temp
set tmp=C:\temp
prompt $p$g
rem win :
```

Conventions

These startup files are primarily for *standalone* PCs, which are not connected to a network. I have no experience with networks, where special drivers may affect memory or other devices.

For typographical reasons, long lines in the following explanations are shortened. Lines that start with REM show options that you may not need, but are included to show what I recommend.

If you are wondering why the C: drive specification is *not* given before the directory name here (apart from in the `PATH` command) it is so that the startup files can be copied to a boot disk needing only one small amendment (see *Boot disks*, p. 55).

I have given two examples:

1. DOS 5 with SMARTDRV.SYS
2. DOS 6, Windows 3.1 or 3.11

Read the text after the examples where I also give some possible lines that are not shown here. If you don't need EMS memory, then my *examples* and *suggestions* for start files are as follows.

The boot process

(A) These are some of the procedures set in motion during the boot process: after a program built into a ROM chip has checked the hardware (memory, ports, video adapter, etc.), information from the ROM BIOS is loaded into RAM memory. The first thing it looks for are the system files, i.e. the hidden files IO.SYS and MSDOS.SYS for MS-DOS.

At this point in the boot process, the computer doesn't know which devices are installed. Every device needs a program (software) that can manage it. Only when this is running can the device become "a member of the family" of PC parts that communicate with the others. If all these processes succeed, the DOS operating system is ready.

DOS 5: If there are no problems up to this point, the PC cannot lock during boot, no matter what lines are in AUTOEXEC.BAT. So, at the very least, it is important to have a CONFIG.SYS file that doesn't make fatal errors.

DOS 6: It is (almost) possible to jump over all of CONFIG.SYS by pressing F5 when you start to boot – see DOS 6 for more details.

At the start of the boot process, conventional memory is empty. CONFIG.SYS is run through twice. The first time, the line `DOS=HIGH` and apparently also `DOS=UMB` is searched for, so it doesn't matter where these line are. If the second run finds HIMEM.SYS, it starts by loading it into conventional memory, followed by EMM386 if it is there, and then DOS is loaded into high memory. Then it goes on loading drivers and other programs into RAM.

CONFIG.SYS

The lines are read in a certain order, no matter where they are placed – except for device drivers. First, the whole group of device(high)= is loaded in the order they appear. Therefore, this order has an important bearing on how upper memory is used. Then (even if the lines do not exist) the lines `files=  fcbs= buffers= lastdrive=` and `stacks=` are read.

Finally, COMMAND.COM, the program that interprets DOS commands, is loaded, possibly via a line `SHELL=`.

Fortunately, drivers can enable a device to perform differently in different circumstances. *You* decide how much of your RAM you want to use for different things. On each line, you can specify one or more *parameters* (a sort of variable) and optional *switches*. A switch can be ON or OFF. A *parameter* usually refers to the object in question, in this case a device. A parameter is a value – not necessarily a number – that can only be one of several options available to the parameter. For example, the parameter *day* can only be one of seven possible values.

This is both good and bad. If you know all the different combinations and their relationships, that's fine. However, as most people discover, trying to memorize them is a shortcut to madness, so this section will try to help you out.

Remember, I have written "`..`" (dot, dot) for the relevant directory instead of "`[path]`."

DEVICE=..\HIMEM.SYS

HIMEM.SYS must be placed *before* EMM386.EXE. HIMEM.SYS is the memory manager for high memory as well as extended memory.

Important: it is not until DOS reads this line that high memory and extended memory exist in a state that can be used – now they have a memory manager that allows access.

By specifying this, all memory above 1088 KB is configured to be available as XMS memory.

DOS=HIGH

The condition here is that HIMEM.SYS is active. `DOS=HIGH` means that DOS places as much of itself as possible in high memory. This is one of the most important features of DOS 5.0 –doing this frees more space in conventional memory to run programs. You are also able to use high memory for other things.

286

If you have a 286 with at least 1 MB RAM, maybe you can use high and/or upper memory, or maybe not. To use upper memory, first you have to have special hardware – which only some 286s have – and then you have to have an upper memory manager. Unfortunately, I don't have any experience with an upper memory manager for a 286. You can find out if your 286 can use high memory. Insert these two lines at the beginning of CONFIG.SYS

```
device=C:\..\HIMEM.SYS
DOS=HIGH
```

and see what appears on screen when you boot. If you don't see it, then type `MEM` when the PC has finished starting. The last line should read

```
MS-DOS resident in high memory Area
```

If not, you'll get the message

```
unable to control A20 line
```

Then try with

```
..HIMEM.SYS/machine:11
```

The number is a machine identification. Allowable values are from 1-14 (see your DOS manual) or experiment, starting with 11, 12 or 13.

I've managed to get it to work on some PCs. If it works, then you've got DOS in high memory, which will make your PC faster because you've got more conventional memory free. Use `MEM` or `MEM/C/P` to check. You should have more memory available for programs. If you can't use upper memory, you must write `device=` instead of `devicehigh=` as well as remove any references to `LH` in AUTOEXEC.BAT.

386

.. EMM386.EXE ..

EMM386.EXE can only be used on a PC equipped with at least a 386 processor. This program is a science in itself with numerous options, not all of which are dealt with here. This long explanation is necessary because it is an important file when it comes to how memory is used.

The program has two basic functions: it can be a *memory manager* for EMS memory and it *provides access* to the upper memory area.

Confusion can arise because EMM386.EXE can be used as a driver as well as a DOS program. We haven't run across this before. It is uncommon to find a device driver that at the same time can also run as a program from the DOS prompt. And the manual ... forget it!

Before reading this line during boot, all RAM above 1088 KB is configured as extended memory. HIMEM.SYS is first of all a memory manager for upper memory but if `DOS=UMB` is specified, then DOS, via EMM386.EXE, takes control over upper memory from HIMEM.SYS.

After you load HIMEM.SYS and EMM386.EXE, it is possible to start to load programs with *devicehigh* and *LoadHigh*.

LH is short for **L**oad**H**igh, which means load into upper memory. LoadHigh and devicehigh should have both been called LoadUpper – the whole thing is totally confusing. If you are trying to load a program/driver into upper memory using a devicehigh or LH and there is no space for it, it loads into conventional memory instead. You don't even get an error message.

DEVICE=..\EMM386.EXE NOEMS

I am going to start with the setup that is relevant for most people. This is the parameter `NOEMS`, which means that some of the extended memory should *not* be converted to EMS memory. `NOEMS` *also* means "create an entrance to upper memory." You use this if you don't need EMS memory, wish to use upper memory and want all your free RAM over 1,088 KB to be used as XMS memory.

Technical corner: With DOS 5, the message I get is `92 KB total available upper memory` and `92 KB` as `largest block`, starting with address `C800` if I have specified `NOEMS`.

I=E000-EFFF

(A) Of no importance to the PS/2. This line is a parameter that is independent of other parameters and can be in the same line as EMM386.EXE. It includes an *address area* in upper memory and depends on the area not being used by anything else – and anything else in this context means something from the system, like the motherboard BIOS.

An area in memory is defined by a start and an end address. As a rule, numbers in base 16 are used, here with only four digits to each – the last 0 is dropped. As an example, the 64 KB in the area from 896-960 KB becomes E000-EFFF. EFFF is the address just before F000. Every area of 64 KB is referred to as a *page*, the A-page, B-page and so on. 640 KB = A000, etc.

EMM386 from DOS 5 defaults *not* to include this area because it is used by the PS/2 to hold BIOS ROM but EMM386 from DOS 6 does include it.

If you want to *ex*clude an area from upper memory (to be sure that no program uses it), specify it in the same way but use an `X` instead of an `I`, e.g.

```
X=B000-B7FF
```

which excludes this area, called the *monochrome region*, the region where the CPU communicates with a monochrome (black and white) video card. It is preferable to *include* this section, rather than *exclude* it. See the comments under *Memmaker* in DOS 6.

Requires EMS memory

Some programs need EMS memory. The most important thing about understanding EMS and upper memory is that most EMS-dependent programs demand that 64 KB *consecutive* UMB be used to make what is called a *page frame*.

On the subject of expanded memory, two parameters will be discussed in detail – `RAM` and `AUTO` (because the manual mentions them only briefly and even then it is wrong as far as the `AUTO` switch is concerned). The DOS 6.0 Help function is also wrong. See page 25, AUTOEXEC.BAT, for where to place `AUTO`.

Let's say that you have a total of 4 MB RAM. You will have 3 MB extended memory after installing HIMEM.SYS – I'm disregarding the 64 KB in high memory for a moment to make things a little easier. EMM386.EXE is able to convert extended memory to expanded memory, so if you want to use 1 MB (= 1024 KB) of your 3 MB extended memory as expanded memory (keeping the remaining 2 MB as extended memory) and want access to upper memory, write

DEVICE=..\EMM386.EXE RAM 1024

`RAM` means "give access to upper memory." `1024` means use a maximum of 1024 KB for EMS memory. If you only give a number, you will be denied access to upper memory.

I=E000-EFFF FRAME=E000

(A). Refer to I=E000-F000 earlier.

Irrelevant for the PS/2. Following LIM EMS specifications versions 3.2 or 4.0, EMS memory uses *page frames*, which I don't describe more fully here. But most programs that need EMS demand a page frame (a "window" in upper memory) that points towards a portion of EMS memory. It is very important that this 64 KB window be available in upper memory – otherwise it is "stolen" from conventional memory, and this is a big portion to give away.

Unfortunately, EMM386 in DOS 5 defaults to a page frame start address of D000, because the IBM PS/2 places the motherboard BIOS ROM here. – if you install, for example, EMS memory with the parameters `RAM 1024`, during boot you will see this as "`starting at address....`"

The parameter `FRAME=E000` determines the start address for the page frame. The example mentioned above makes better use of upper memory.

EMM386 in DOS 6 has been improved. It would seem that MEMMAKER tests to see if E000-EFFF (the E-page) is free, and if it is, it chooses E000 as the start address. Fine. At any rate, that is how it works on my PC.

FRAME=NONE

(A) It’s possible that your programs that use EMS memory don’t need a page frame (though this applies to very few programs that use EMS memory).

If you would rather take advantage of the 64 KB in upper memory, by letting Windows or other programs use it, you can specify this parameter. If you use DOS extended programs such as AutoCad386 or Lotus 1-2-3 ver. 3.x, which expand memory above DOS’s normal limits themselves, you should check with the manual to see whether or not you will benefit by writing this.

This is undocumented in DOS 5.0, while DOS 6.0 mentions it, and adds that it may cause some programs not to function properly. The box at the bottom of the page shows some example parameters for EMM386.

DOS 6: see EMM386 p. 44.

(Enough about EMS and EMM386.)

DOS=UMB

means that DOS takes over the handling of all upper memory. DOS requests HIMEM.SYS to pass over control of all upper memory. You can write the two lines as one:
`DOS=HIGH` and `DOS=UMB` as `DOS=HIGH,UMB` but as they are two separate commands, I have chosen to write them as two separate lines.

REM DE..IGH=\..\CDMKE.SYS /D:MSCD01

This is a driver for a Panasonic CD-ROM drive that I activate when I need it – it uses 11 KB. The name given after `/D:` identifies the CD-ROM drive to the MSCDEX program that is called in AUTOEXEC.BAT. See MSCDEX on p. 24.

SMARTDRV.SYS 512 256

This line is only relevant if your disk cache is the not-so-good SMARTDRV.SYS from *DOS 5*, that is, you do not have Windows 3.1 or DOS 6. If you don’t know what a disk cache is, read the general description *SMARTDRV disk cache* on p. 28, then return here.

The numbers are the amount of KB for the cache’s initial size and minimum size. You are free to choose whether you want to state just the first number, both numbers or none at all. The optimum numbers depend on the available amount of extended memory. The default for the first number is 256.

The last number is relevant for Windows. Certain programs have the ability to change the minimum size and even give it a value of 0 in order to use this area in memory themselves.

To avoid this you must state the minimum by setting the second number. I am assuming that you have not installed expanded memory. The above is an example with 2 MB total RAM and `NOEMS` at EMM386.EXE. In other words, 960 KB available XMS memory. When you have used 512 KB for SMARTDRV, you are left with 448 KB available for programs that can use extended memory. If you know that this is not the case for your programs, try and set the first number higher and see if that speeds up your system.

If you have 4 MB total RAM, write `1024 512`. You don’t need to use a disk cache – it is, however, recommended.

(SMARTDRV only became an acceptable product with the introduction of ver. 4.0 from Windows 3.1, and became worth using with version 4.1 from DOS 6. If you have SMARTDRV.EXE, you are much better off. You shouldn’t load it from CONFIG.SYS but from AUTOEXEC.BAT. You can still use SMARTDRV.EXE 4.0 even if you only have DOS 5, see also p. 29.)

```
device=c:\..\emm386.exe ram 1024
device=c:\..\emm386.exe i=e000-efff frame=e000 ram 2048
device=c:\..\emm386.exe i=e000-efff x=b000-b7ff noems
device=c:\..\emm386.exe i=e000-efff noems
```

..RAMDRIVE.SYS 2048 /E

Relevant if you have Windows 3.1, a minimum of 8 MB RAM and use programs that work intensely with temporary files. Windows and many Windows programs do this, e.g. CorelDraw.

It creates a *RAM disk* (IBM usually calls it a virtual disk), i.e. a temporary hard disk that exists only electrically while your computer is running. During the boot procedure, you will see the next available drive letter, at the end of the `virtual disk` line. The number specifies how much RAM is set aside to behave as a disk. Here it is 2048 KB of extended memory – indicated by the `/e`.

If you want to use EMS memory (you may have a card with expanded memory), use `/a`.

The risk with this is that if the computer hangs or there is a power failure, all data in the RAM drive is lost. The advantage is faster access to data when you work intensely with temporary files. It requires that you enter in AUTOEXEC.BAT the line

```
SET TEMP=D:\
```

(or your next available drive designation).

..DISPLAY.SYS CON=(,,1)

Is the driver for the console. The *first* parameter, which refers to the type of monitor/graphics adapter, need not be stated since DISPLAY.SYS checks the video adapter automatically.

The *second* parameter refers to the number of the codepage for the console supported by the hardware. In my experience, if you use `mode con cp prep` and `mode con cp select` in AUTOEXEC.BAT, where this codepage is chosen. If your hardware (video card) doesn't support one of the codepages for your country, you must give a number. To see this, type `mode con` in DOS.

The *third* parameter (here, 1) is the number of codepages needed for which space has to be reserved in memory (but still only for CON and not any other device). The default is 1. If you only want to use one codepage, e.g. 850, write 1 at the end.

All this is necessary to be able to use `mode con cp prep` and `select` in AUTOEXEC.BAT.

For two codepages: If you choose 2, you will be able to shift between codepages at the end, e.g. 437 and 850, providing you have prepared them (see `mode con ..`) and loaded NLSFUNC in AUTOEXEC.BAT. This allows you to switch between codepages with the command `CHCP 437` or `CHCP 850`. DISPLAY.SYS occupies 8.1 KB.

See buffers on page 24 to see how many buffers can be loaded into high memory.

...SETVER.EXE

Some programs – typically older ones – need access to a certain DOS version number. SETVER lies to these programs about the version number. Type `SETVER` to see this. Requires 432 bytes.

..GMOUSE.SYS

Loads a mouse driver for DOS programs. Windows has its own driver but this is only for Windows programs. This example is for a Genius mouse. Your mouse driver may be called something else and be located in a different directory.

..ANSI.SYS

Is a driver for screen characters, cursor movement and defining the keyboard keys. Necessary for DOS if you want to use something other than the default 25 lines of 80 characters, changing the colors on the DOS screen, etc.

COUNTRY=044,437..

In this example, I have chosen an English user who wants to use codepage 437 as the active general codepage. The first number, in this case 044, sets English formats for time, date, currency symbol, sort order in files and characters used in directory and file names.

The second number sets the active codepage, the general active codepage for all devices that are supported by it.

If you leave the second parameter blank, you will automatically choose, during boot, the default codepage (see the manual for the country command or with DOS 6.x, type at the DOS prompt `help country` and select `notes`) and not the alternative codepage. On an “English-speaking” PC, the default codepage is 437 with 850 as an alternative. In this example, the default codepage is selected.

For more, see the Codepage section, p. 28.

BUFFERS=5

Is a mini cache programme (see page 28 for an explanation of disk caches). If you are using SMARTDRV I would suggest that you set the number of buffers low, to 6 for instance. Every buffer uses about 532 bytes. If you do not use a disk cache program, then you should experiment a little. Try values of between 6 and 20 and see which one works best on your PC. You will not have any problems, just a slightly faster or slower computer.

The majority of buffers should, according to Microsoft, be loaded into high memory if there is room there. Whether I used `MEM/C/P` or `MEM/D/P`, I could not see any evidence of this. I discovered the cause later: if DISPLAY.SYS is loaded (and this happens in all non English-speaking countries to allow access to country-specific letters and symbols), then something happens in high memory. An educated guess: the codepages that have memory reserved for them via DISPLAY.SYS take up space in *high* memory.

Experiments showed that if DISPLAY.SYS is loaded with the last parameter (number of codepages) set to 1, the result is that only 27 or fewer buffers can be held in high memory, with 512 bytes used in conventional memory. With 28 buffers, *all* of them are pushed down into conventional memory. Another peculiarity: with buffers=24-27, COMMAND.COM uses 5 K, otherwise it uses only 3 K. If you are loading a single codepage, the most efficient way is to specify buffers=23.

If the last parameter in DISPLAY.SYS is set to 2, then only eight or fewer buffers can be placed in high memory. With nine buffers, things go wrong. With buffers=6-8, COMMAND.COM fills 5 K, otherwise it only fills 3 K. With two codepages specified, the best setting is buffers=5. Luckily, there are only a very few users who need so many buffers, as nearly everybody uses a disk cache like SMARTDRV, but *it is a mistake that the manual or Help does not explain this*. Almost certainly, the Americans have never even discovered this problem, as they do not need DISPLAY.SYS. (IBM's PC DOS 6.1 and 6.3 behave in exactly the same way).

If you want to test things out, then use `MEM/D/P` after every boot, and look for BUFFERS. You can also use the undocumented `MEM/A`, which also tells you about high memory.

FILES=40

The default (if the line isn't included) is 8. Valid: 8-255. Specifies how many files can be open at the same time. A file is "open" when it is in use or being read. Today, many programs need to have a lot of files open at the same time. If you use Windows and run several programs simultaneously, you require a higher amount.

Some space is taken up in memory for this but not much. Try 30, 40 or 50 and see how it works. I wonder why Microsoft doesn't provide a program that shows the number of open files.

REM FCBS=X

File Control Blocks. Certain older programs, e.g. SideKick, access files by means of FCBs. The value *x* states how many of these blocks DOS can have open at the same time. The default is 4.

If you don't know whether or not your program requires FCBs, try setting the value at 1. If you get no error messages, you will have saved a little memory.

REM STACKS=9,256

The numbers given here are only examples. The default (if the line isn't included) is 0.0 for the original IBM PC and 9,128 for all others. Specifies how much memory is set aside to handle *hardware interrupts*.

An interrupt is when there is a "...telephone call for Mr. DOS" while it is doing something. If you press a key while DOS (and hence the processor) is in the middle of a task, it will be taken as a call that can't be ignored. So DOS will temporarily place what it is doing on a "shelf" in what is called a stack (a little buffer) while it handles the interruption. When the interruption is over, DOS returns to what it was doing.

Examples of interrupts include clicking or moving the mouse, hard disk activity or an incoming fax. Many interrupts occurring at the same time can use up the stacks reserved here; the solution is to increase the number.

The first number gives the number of stacks. Valid numbers here are 0, or from 8 to 64.

The second number specifies how many bytes are set aside for each stack. Valid numbers are 0, 32, 64, 128, 256 or 512. If you have extra cards for a scanner, fax modem, modem or similar device, and have ever had an error message *stack overflow*, you can try raising the values and see if it helps. Try 9,256 – 9,512 – 10,128, etc.

If you don't have an original IBM and you don't have this line in your CONFIG.SYS, you waste memory unnecessarily. If you want to aim for the optimum setting, try `STACKS=0.0`. It may just be that none of your programs requires these stacks. If you encounter problems, then write `STACKS= 9,128` – which is the same as omitting the line. If you need a higher number, it is already there so you can change it more easily.

It does, however, use some *conventional* memory, only a tiny amount, but it's worth mentioning.

BREAK=ON

Increases the number of times DOS checks to see if the user has pressed Ctrl+C or Ctrl+Break (pause key) to stop a program. However, certain programs are written so that these keys have no effect. `break=ON` means here that you *can* break a program with Ctrl+C or Ctrl+Break. `break=ON` has no effect on the use of RAM.

SHELL=.. /p /f /E:1024
SHELL=..\DOS\..   /p /f /E:1024

Some people place COMMAND.COM in the root and don't have a line stating `SHELL=`, which is OK. Others place COMMAND.COM in C:\DOS together with this line, showing where it can be found.

The first is the easiest way as by default the boot process looks for COMMAND.COM there but the line above can specify where it is.

Placing it in C:\DOS is also reasonable because you seldom work in the DOS directory and therefore you're not likely to delete COMMAND.COM by mistake. *If* you delete COMMAND.COM in error then you'll lose your command interpreter. Solution: boot from a floppy and `copy` it from the floppy to the hard disk. You can now boot from the hard disk.

`/p` means that you have specified the permanent command interpreter (you may use another but that is not dealt with here).

`/f` means that the option `Fail` is chosen if you get the error messages `Abort, Retry or Fail`, which typically happen when a program tries reading a diskette drive and finds no diskette.

`/E:1024` – another very important purpose of this command is to expand the *Master DOS environment* to the number of bytes specified at the end of the line. This is a sort of bulletin board in memory where DOS and other programs save and retrieve certain internal information and variables.

Some programs require that a variable be defined in this environment, and during installation of such a program, a line in AUTOEXEC.BAT is inserted or recommended, e.g. `set lib=C:\QB`. Try entering `SET` and `HELP SET` at the DOS prompt; it will give you an idea of what it is all about.

If a line with `SHELL=` isn't included, DOS 5 sets aside 160 bytes and DOS 6 earmarks 256 bytes.

If you get an error message *Out of environment space*, you can try increasing the number in this line. I have never needed it to increase the amount of bytes, but Microsoft recommends setting the number to 1024 or 2048 for Windows 3.1. It is a question of a very few bytes, so you use very little conventional memory, 1 or 2 KB.

If you have a long path statement, with around 127 characters, then 127 bytes are used for this so it can be relevant to set a number here. Most people will find that 512 or 1024 works fine.

DOS 6.0: MEMMAKER inserts this line if it isn't found, although without `/e:1024`

Other possible lines in CONFIG.SYS

Even this guide has its limits!

REM LASTDRIVE=X

This stipulates the maximum number of drives you can access. The default is the next available drive, which for most people is drive D. You waste memory if you write Z without needing it.

Network users: certain network programs attempt to create drive names beyond the maximum, in which case Z does not work.

FASTOPEN

If FASTOPEN is specified in your CONFIG.SYS, I recommend that you delete the line. The idea of having fast access to files is attractive, but users have reported damaged files as a result of using it. This file is also included in DOS 6.

AUTOEXEC.BAT

The file AUTOEXEC.BAT contains DOS commands that are executed when the computer starts up. It is essential that the file be stored in the directory from which the PC boots (usually `C:\`). It should be noted, however, that *the file need not exist at all*.

@ECHO OFF

Prevents commands from being shown on the screen while they are being executed. The @ at the beginning of the command prevents even the words “Echo Off” from appearing. Try placing `REM` at the beginning of the line; that makes the booting sequence “visible.”

PATH C:\BAT;C:\DOS; . . . C:\UTI;

Semicolons separate the different directories. The maximum number of characters is 127! Type `PATH` at the DOS prompt to check what your path currently contains.

Many programs want to add one or more directories to the path during installation. If you agree to this, you might see a line added with `%`. This means “add this directory to the path.” Edit the path yourself if you want to include it and then delete the line with `%`.

If you leave out `C:`, then the path will only work when you are on drive C, which is not so clever. For example, you would not be able to carry out an `xcopy` command if you are on the A: drive.

REM LH MSCDEX /D:MSCD01 M:20

The MSCDEX program allows you to use the CD-ROM drive. The name after `/D:` must be the same as that given in the driver line in CONFIG.SYS. It is very important that this line come *before* the line activating SMARTDRV so SMARTDRV can act as a *read-cache* for the CD-ROM drive. For more information, see SMARTDRV on page 28.

When SMARTDRV starts, it checks to see if MSCDEX is running. If so, SMARTDRV sets a default value read-cache for the CD-ROM drive.

If you want to see how much memory is allocated, then add `/V` to the line and this figure will be shown when booting. `/M:20` stipulates the number of sector buffers.

C:\..\SMARTDRV.EXE

SMARTDRVE is shown in this example without parameters for starting point and minimum sizes. SMARTDRV checks for available XMS memory and chooses the sizes itself, which works fine for most users. To ensure that SMARTDRV is loaded into upper memory, it is a good idea to place it early in AUTOEXEC.BAT as it takes up 26 KB memory. See *SMARTDRV Disk cache* on p. 28 for more.

MODE CON CP PREP=((850)..
MODE CON CP SEL=437

`MODE CON` is a setting (a state to be in) for the console. CP is short for codepage. `Prep` means prepare. The file EGA.CPI contains the graphics characters that are displayed on the screen.

I have chosen examples allowing for two codepages, which is only possible if space has been prepared in memory first (see `display.sys`). The first line prepares for the console to use two different codepages. The second line chooses the codepage that the console is to use, which means in practice how ASCII values are shown on the screen. `MODE CON` gives information about

1. the (optionally chosen) hardware-supported codepage for CON – this depends on the second parameter in the line with `display.sys`
2. the prepared codepage(s) – controlled by `MODE CON CP prepare= ..`
3. the chosen codepage – controlled by `MODE CON CP select=..`

If you have only reserved room for one codepage in the line with `display.sys`, you can prepare and choose only one codepage, 437 or 850. If you have chosen two, then in the first line write
`..prep ((437 850) ..`
See the manual for the difference between the pages; typically it will be symbols like the copyright sign © and graphic symbols used for drawing boxes.

Keyb.com..keyboard.sys

Must come after `mode con cp select`. Installs or configures the keyboard for an English layout. As the second parameter isn’t stated, the codepage chosen with `mode con cp select =` is the one selected.

If the second parameter is specifically stated, it must be identical to the one chosen by `select`. Takes up 7 KB.

Rem LH ..nlsfunc.exe

loads a program, making it possible to change between codepages using the `CHCP` command, e.g. `CHCP 437` or `CHCP 850`. Only relevant if you want to be able to change between two codepages, e.g. 437 and 850 in Great Britain.

REM LH ..DOSKEY.COM

DOSKEY is a handy little program that remembers your most recently used DOS commands. Use arrow up and arrow down to browse them. Only relevant if you write many (long) DOS commands. Takes up 4 KB.

LH C:\DOS\SHARE

SHARE.EXE manages which files are “open” so that two programs can’t open the same file. I have tried it with different programs but not all give a warning, so you can’t be sure that all programs can use it.

It only takes up 6 KB and I recommend it, especially if you are running Windows. For WinWord 6.0, Microsoft suggests this line:

```
LH C:\DOS\SHARE /L:500 /F:5100
```

Where `/L:500` gives the number of “locks,” defining how many locked files can be managed at the same time. `/F:5100` specifies, in bytes, how much memory should be used to make sure two programs cannot use the same file simultaneously.

set tmp=C:\temp
set temp=C:\temp

forces programs, such as Windows, to use the specified directory for saving files *temporarily*.

It sometimes occurs that your PC crashes, i.e. stops responding, and you have to reboot. If this happens, and a program wasn’t closed properly, temporary files may be left behind. They are easy to find (and delete), especially if they are in this directory.

The file name usually starts with a ~ symbol and often has TMP as extension. Delete these files regularly from your hard disk but only when you are at the DOS prompt with no other programs running.

rem SET TMP=D:\
rem SET TEMP=D:\

If you have installed a RAMdisk in CONFIG.SYS and also want to use it as a drive (disk), where temporary files can be stored, you can stipulate the RAMdisk – in this example drive D – as this drive. If the next free drive is another letter, of course, you must use that letter. During boot, RAMDRIVE.SYS will show the drive chosen. Remember to enter both lines.

PROMPT $P$G

Almost everyone uses this prompt, which shows you which drive and directory you are in (with DOS 6 you do not need to insert it; DOS 6 does it for you). `C:\>` is called the DOS prompt because the computer shows that it is ready and is “prompting” you to write something. The `PROMPT` command can also be used to redefine keys or give you other screen colors. The following prompt command requires that you have installed ANSI.SYS from your CONFIG.SYS file, as described earlier. If you have, try entering

```
C:\>PROMPT $e[0;1;37;44m$P$G
```

The square parenthesis can be produced by pressing Alt+91 (if you don’t have it or can’t find it on your keyboard). Then enter `CLS`. This will give you a blue screen with white letters. If you like this better, you can enter the above in your AUTOEXEC.BAT file instead of `PROMPT $P$G`. The whole screen will be blue when you work in DOS. If you don’t like it, either re-boot or type
`PROMPT $e[m$p$g`.

Rem ..gmouse.com

I used to have a Genius mouse with the driver C:\UTI\GMOUSE.COM. By removing `rem`, I could load GMOUSE.COM into upper memory. Generally, though, I don’t use a mouse in DOS programs as I find the keyboard faster.

Mouse drivers for *DOS* programs are often accompanied by one file with the extension SYS *and* one with the extension COM. The SYS file can be loaded in CONFIG.SYS. The COM file can be loaded in AUTOEXEC.BAT.

Windows has its own mouse drivers for different mice, and they work in *Windows* programs. If you want to use a mouse in DOS programs under Windows, you have to install the mouse driver before you run Windows. To be able to use the mouse in a DOS window (not a full screen), the driver must be minimum ver. 8.20 from Microsoft, or compatible. You might need to add this line in SYSTEM.INI:

```
[NonWindowsApp]
MouseInDosBox=1
```

`REM ..EMM386.EXE AUTO`

Only relevant in DOS 5 if you have installed *expanded* memory. This parameter can be specified when EMM386.EXE is loaded as a program, that is to say from a DOS command. It does *not* work in CONFIG.SYS. *Auto* means that the amount of KB set aside as EMS memory in CONFIG.SYS is only used as such if a program asks for it, otherwise it is used as extended memory. But once used as XMS, it won't return to EMS; it doesn't switch back automatically.

`rem WIN :`

Many people use the last line to start a program up, typically a menu or Windows. Here "Win space colon" – skips the advertizement as Windows starts.

`set winpmt=Type EXIT and press ENTER to return to Windows$_$_$p$g`

is a good idea, and should be written on one line. `Winpmt` is short for Windows' prompt, and this line means that when in a DOS box you will be reminded that you are running a DOS box in Windows (if you forget and reboot from your DOS box, it can have unfortunate consequences). `$_$_` has the effect of entering two blank lines (Enter, Enter).

Codepages

(A) Advanced. This relates only to DOS codepages. Windows has its own codepage, which is defined in Windows Setup.

I have spent a long time experimenting, reading the manual, writing `chcp, keyb` and `mode con` during boot and in several different places in AUTOEXEC.BAT.

On different PCs, I have seen many different versions of the relevant lines in the startup files. Few people seem to have problems, even if there is a difference between the active cp (type `chcp)` and the codepage for the console (type `mode con)` – but some do, especially during Windows setup, if they do *not* give other parameters in the country command.

It doesn't help matters at all (and here I criticize Microsoft) that information is not given to every country about codepages in its own language.

First, the DOS 5 manual and the screen messages returned by the above commands (those that have anything to do with cp) are the most inconsistent (both literally and in the information they provide) that I have seen for a long time. The DOS 6 manual is better but not where it is important.

It would appear that Microsoft has chosen `chcp` to be the "king" of all commands because at one stroke it can change *all* cp definitions for *all* devices. My experience is that using the second parameter in the `country` command is just as good.

It is utterly ridiculous that you have to insert several complicated lines in a certain order in the startup files simply to define a codepage or to be able to swap between two. I have given up trying to get an explanation from Microsoft for why they have made the whole thing so complicated.

As I have already mentioned, a potential problem does exist if you don't stipulate the second parameter in the `country` command in CONFIG.SYS, which is why I mention it here.

I know this is of little practical significance for most readers but maybe with more international data communication in the future, it will become more important.

SMARTDRV disk cache

Generally

There are several different disk cache programs on the market but I discuss only SMARTDRV here, as it comes as an extra program with DOS. A cache (pronounced cash) is a buffer, a link, an intermediary storage place between the CPU and the hard disk. A primitive form of cache is `buffers=` in CONFIG.SYS.

SMARTDRV by default uses part of extended memory for its buffer. When the CPU needs data, it checks the cache first to see if it is there (a hit). If so, it will be read from cache, which can be 100 times faster than if the data is not in the cache (a miss) and the CPU has to get it from the hard disk. This is called *reading* from cache.

If the CPU is told, for example, to save a document, it will first be saved in the cache. This is *writing* to the cache.

If you don't know very much about this program, or don't know whether or not it is installed on your PC, write

```
C:\>SMARTDRV/S
```

This will not do any harm. If the program is not installed, it will just display the different options, or parameters, available.

If it is installed, it will show the start `cache size` in the first line and in the second line a minimum `cache size while running Windows`.

It also shows how many hits and misses SMARTDRV has had. The point is, of course, to get as many hits as possible, so you can try this command to see how things are going – even while you are running Windows. Finally, it shows the drives where read and write caches are enabled.

SMARTDRV.EXE

Version 4.0 and 4.1: The following applies to version 4.0 (from Windows 3.1) and 4.1 (from DOS 6.0).

SMARTDRV loads itself in upper memory if possible. If a start size and minimum size are not stated, SMARTDRV will find out how much *XMS memory* to use, which works fine for most users. Drive C will be the read and write cache drive by default. The start value, or size, can also be called the disk cache size, though it only applies while running DOS programs.

With 4 MB of RAM installed, 1 MB is occupied under DOS and a minimum of 512 KB while running Windows. When Windows is loaded, it allocates extended and available upper memory for different purposes (Windows has its own memory manager.) Windows and SMARTDRV cooperate in using memory efficiently, which is why you can state a minimum size under Windows. The path line in your AUTOEXEC.BAT must include the directory in which you have SMARTDRV.EXE.

`SMARTDRV 1024 1024`

This is an example that shows a start size and a minimum Windows size given in KB. You have to experiment to find out what is best for your own configuration. I suggest that you run with one setup for a while, change to another setup, change back again and see if you can feel any difference. Objectively, there may be a difference but if you can't feel it, it can't be that big. It is a matter of finding the right proportions.

You will have to boot between tests in order for the new sizes to become effective.

The following commands apply to all versions

`SMARTDRV/?`

Lists the different parameters.

`SMARTDRV C+`

Turns *read/write* cache on drive C *on*.

`SMARTDRV C-`

Turns *read/write* cache on drive C *off*.

`SMARTDRV C`

Turns *write* cache on drive C *off* but leaves *read* cache *on*. If you have several drives on your hard disk, you may wish to switch off the write cache on one of the drives. This can be entered at the DOS prompt but I recommend waiting until you are sure that all activity on the hard disk has ceased to be sure that no data is left "hanging" in the cache, which leads us on to:

SMARTDRV/C

Forces SMARTDRV to save everything from the cache that's not already saved to disk. It is always good to give this command (in BAT files for example), especially if you or your PC change quickly from one program to another.

In the SMARTDRV 4.1 section, the DOS 6.0 manual warns: "Write **SMARTDRV/C** in DOS before the PC is switched off to be sure that SMARTDRV has saved all your data to the hard disk. After all activity on the disk has ceased you can turn off your computer without risk."

First tip: always exit Windows before you switch off your PC. Second tip: *wait until any hard disk activity has ceased* before switching off your PC.

SMARTDRV 4.2: Many users complained about losing data because, in all good faith, they had switched off their PCs too soon (before the cache had emptied to disk). As a result, Microsoft – keeping a low profile and without any public announcement – released version 4.2 before DOS 6.2, which, like earlier versions, still has a write cache as default but writes to disk *before* returning to the DOS prompt after closing a program. This facility is new! With this, the user is certain not to lose data – although a small price is paid in the form of reduced speed.

SMARTDRV 5.0: Comes with DOS 6.2 and Win 3.11, and if it is a new installation, switches off the write cache and writes directly to disk. If the installation program finds that SMARTDRV is using a write cache, it keeps it. Ver 4.2 and 5.0 contain two new switches:

`/X` switches the write cache off on all drives, in other words, writes directly to disk.

`/N` causes SMARTDRV to wait – just like in versions 4.0 and 4.1 – for the CPU to have a free moment before writing to disk.

Ver. 5 supports caching on a CD-ROM drive. Assuming that you only have one drive on your actual hard disk, it is the surest and simplest way to insert the following in AUTOEXEC.BAT:

Version 4.0 and 4.1

```
C:\..\SMARTDRV C
```

(This writes directly to disk.)

Version 4.2 and 5.0:

```
C:\..\SMARTDRV
```

(This writes to the cache, but to disk before returning to the DOS prompt. The Win 3.11 installation inserts `/X`, i.e. no write cache at all.)

In all circumstances, a read cache is used (which is the most important one for making your PC run fast) and the cache writes to disk before you see the prompt. By typing SMARTDRV/S at the prompt, you can see which version of SMARTDRV you have.

Double buffering

Is, as the name suggests, a double buffer, which is required by certain hard-disk controllers. The AT bus doesn't need it, but some SCSI and some ESDI and MCA controllers do. The line in CONFIG.SYS

```
device=..smartdrv.exe/double_buffer
```

makes these controllers compatible with the type of memory created by EMM386.EXE or by Windows running in enhanced mode. The driver uses 2 KB of conventional memory and *cannot* be loaded into upper memory. A PC with a controller of this type needs this line in CONFIG.SYS. And how do you find out? Insert the line in CONFIG.SYS and SMARTDRV in AUTOEXEC.BAT, boot and write `smartdrv /s`.

If, in the "buffering" column it says yes, your PC needs the line in CONFIG.SYS. If it does not, then you can delete the line again.

The hard disk

Many factors influence the speed at which data can be moved from the hard disk to the CPU for processing. Let's look at some of them.

The files on the hard disk are arranged in a certain way and in a certain order. If we imagine the hard disk as a circular plate with its reading head placed in the middle, we can compare it to a gramophone record. The files start in the middle, one after the other in a long spiral chain, like the music in one long groove.

Imagine that the reading head is the record player's pick-up – placed in the middle. When the head is asked to read a file, it looks in a table, the **F**ile **A**llocation **T**able, where it finds the file's physical location on the disk. The head then moves outwards a bit and turns the disk until it comes to the beginning of the file.

But there is a big difference between a gramophone record and a hard disk: files on your hard disk come and go. They get deleted every once in a while, and new ones are created.

Imagine that you have placed 100 files on your hard disk without having deleted any. The 100 files are placed nicely one after the other. Now you delete three files with these sizes: the first 50 KB, the second 100 KB and the third 80 KB. That leaves three available and unequal spaces on the disk.

Then you save or copy a file. If the file is larger than 230 KB, DOS will place the first 50 KB on the first empty space, the second 100 KB on the second space, the next 80 KB on the third space and the last part of the file will be placed after all the others. The file has been divided or *fragmented*.

Next time the file is to be used/read, the reading head has to go on a long trip around the hard disk, and that takes time. I have purposely simplified something here: in reality the hard disk is spinning constantly; this is necessary to attain the speed at which the hard disk is read. Floppy-disk drives don't spin until they are told to do so.

You have probably worked out by now that files that are nearest to the center of the disk and are contiguous are read fastest. The most important thing is that they are contiguous, i.e. they are in one piece. Where they are on the disk is less important (though the longer the seek time, the more important it is). In the next section, we will look at a program that can tidy up the data on a hard disk, joining together the file fragments for faster reading.

The time it takes for the reading head to access the beginning of any file is called the "*average seek time*" or *access time*. At present, the most popular hard disks on the market have an average seek time of 10-15 milliseconds. Trade advertisements usually give only this specification about a hard disk – as if it were the only thing that indicates the quality. It *is* important but other factors such as a disk cache (software- or hardware-based) are more important. *PC Magazine* uses a test that gives a hard disk's "throughput," an average for data transfer that takes many factors into account.

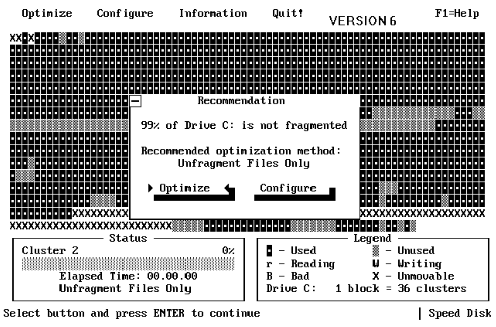
Windows programs work with much larger amounts of data than was usual a few years ago.

Let's have a look at something you can do to get your PC to run faster. You can make the files contiguous and place the ones you use most as close as possible to the physical center (the logical "start") of the hard disk as possible – and now we shall look at a program that can do this.

Disk optimization

Many programs on the market optimize or *defragment* the hard disk. DOS 5 does *not* include one. DOS 6 comes with a reduced version of SPEEDISK.EXE, which it calls DEFRAG. The one I'll be discussing here is the SPEEDISK.EXE version 6.0 from Norton Utilities (a collection of various "housekeeping" programs not traditionally included with DOS).

I wrote this section when I had a hard disk with a seek time of 18ms. Now hard disks have seek times in the region of 10ms or less. Disk optimization has a *greater* effect the *slower* your hard disk is.



```
Recommendation
99% of Drive C: is not fragmented
Recommended optimization method:
Unfragment Files Only
Optimize   Configure

Status
Cluster 2                    0%
Elapsed Time: 00.00.00
Unfragment Files Only

Legend
■ - Used          ▒ - Unused
r - Reading       W - Writing
B - Bad           X - Unmovable
Drive C:   1 block = 36 clusters

Select button and press ENTER to continue          | Speed Disk
```

No matter which disk-optimization program you use, *make a backup of all your important files before you start using the* program. I have never had problems with SPEEDISK but things can go wrong. A power outage in the middle of your optimization process could be a disaster ...

I recommend that you do not have a disk cache active. If you have one in one of your startup files, put it temporarily out of action by writing `REM` in front of the line that loads it, and reboot. Microsoft does not recommend this for SMARTDRV. However, I've "forgotten" to put SMARTDRV out of action a couple of times – with no detrimental effect. But better to be safe than sorry.

1. You must be at "DOS level," i.e. straight after a boot with no programs running. If you use DOS 6, you can press F5 while the message `STARTING MS-DOS` is displayed, and then you can be sure that no disk cache is active.

2. Another, and *more important*, point is to check drive C (or the drive you want to optimize) with `Chkdsk/F`, which should repair any faults. If you want to check a compressed drive (DblSpace, Stacker or SuperStore), these usually have their own check program that replaces DOS's `Chkdsk`. If you have DOS 6.2, SCANDISK will start. If you are using DBLSPACE, use DEFRAG.

The most important thing an optimization program does is to bring together the fragments of each file, i.e. every file will physically be in one long run. If you've never used such a program before, you will probably be very (positively) surprised by the improvement it gives you.

If you don't feel like experimenting with the order of directories and files and so on at first, just satisfy yourself with an optimization that brings files together in one whole piece. With SPEEDISK, this is called "Unfragment files only."

You can adjust the program by stipulating a variety of parameters. For instance, you can decide the sorting sequence of your directories, and within each directory, you can choose to sort files by all or any of the following criteria: name, extension, size and time, in either ascending or descending order. Or you can simply choose "merely" to optimize the files that are fragmented, and nothing else.

It is certainly best to plan before you start, which is something you can do more easily after you have had a little experience with the program. You will also find that it is only after you have worked with your PC for a while that you can derive most benefit from a disk-optimizsation program. But when that time comes, you will be able to work out which programs you use most, and which directories contain these programs' files.

Next, you need to know which programs read and write to the hard disk the most. If you are not sure, try listening to the hard disk while you are working with different programs. Windows and most Windows programs use the hard disk intensively.

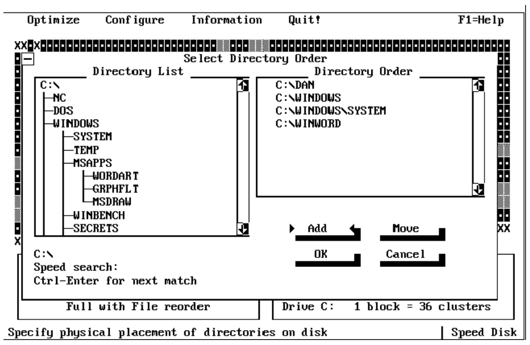
My experience is that word processors, even word processors running under Windows, don't read and write to the hard disk so much. I haven't worked much with spreadsheets but big ones are certainly active users of your hard disk, and databases put even more demands on it. Programs that work a lot with graphics are demanding.

You will have to assess the importance of different, though related, factors: e.g. how often you use the program and how disk-active it is. I place the directories for the most disk-active programs first if I use them regularly and fairly often, and move the rest according to the tasks they perform.

Let's look at SPEEDISK ver. 6. There's no need to be nervous, you don't have to begin optimizing yet. We're just looking at the possibilities. If you have a mouse, activate it. You start the program by typing `SPEEDISK` at the DOS prompt. The first thing you will be asked is which drive you want to optimize, which will usually be C. The program checks the directory structure, and examines how fragmented the disk is. If relatively few files are fragmented, it will suggest that you simply optimize these files. You must *not* press Enter – if you do, then press Esc and answer Cancel. You can always stop an optimization by pressing Esc.

Go ahead and choose configure, using the right-hand arrow and enter, or click with the mouse. The program shows the disk, divided up into small sections. You can see, by referring to the explanation elsewhere on the screen, which sections are used and which are empty. Crosses are files that may not be moved (immovable files) such as DOS system files or perhaps a Windows *permanent swap file* – described in the section *Windows*. SPEEDISK does *not* move these files.

Only the most important options are explained here. The most important choice is the method that SPEEDISK uses. Under Optimize choose the option Optim. Method. Using the keyboard, move round with the arrow keys, and mark or remove a checkmark with the space bar.



```
Select Directory Order
Directory List                 Directory Order
C:\                            C:\DAN
├NC                            C:\WINDOWS
├DOS                           C:\WINDOWS\SYSTEM
├WINDOWS                       C:\WINWORD
│ ├SYSTEM
│ ├TEMP
│ ├MSAPPS
│ │ ├WORDART
│ │ ├GRPHFLT
│ │ └MSDRAW
│ ├WINBENCH
│ ├SECRETS
C:\
Speed search:
Ctrl-Enter for next match
Full with File reorder         Drive C:    1 block = 36 clusters
Specify physical placement of directories on disk        Speed Disk
```

Full optimizes all files but doesn’t rearrange the order of files and directories. Full with File reorder is the one I use myself. This places files in the order chosen using Directory Order under Configure (described below). Unfragment Files Only is the quickest way to get contiguous files.

Go to Configure and choose Directory Order. It is a little difficult to use your keyboard here. On the left is your directory structure. On the right is the order you have asked SPEEDISK to place your directories on the hard disk, if this option is available with the method chosen.

If you choose the method Full with File reorder, the directories that are in Directory Order are moved to the “front” of the disk.

The Tab key moves between windows, and the arrow keys move the marker. Note that you can mark a directory on the right and then change its priority. Enter works differently, depending totally on what you are doing. This is confusing at first, so it is easier if you use the mouse. Experiment a little to find out how to list the directories you have chosen.

Choose Save Options to save what you have chosen. By choosing File Sort you can choose how files in every directory should be sorted. If you have a lot of files in one or more directories, or if they are very large, this can be a significant point; otherwise it is of less importance than the order of directories. On the other hand, there are some more important possibilities under Other Options. I recommend that you choose Read after Write, which means that the program checks that data has been moved intact. If you do not use this, then the optimization process goes much faster, but you risk ending up with the occasional corrupted file, which means you will have to reinstall the program it belongs to. Save this option by using Save Options.

On a PC from 1991, a full optimization of a 90 MB hard disk can easily take a couple of hours. When you have finished optimizing, it can be interesting to see where individual files are located on the disk. You can do this by using Walk Map on the Info menu. Again, here is a situation where it is good to be able to use the mouse.

If you save new files just after a disk has been fully optimized, they will be placed last on the disk. This might not be what you want, so here is a little tip. The idea is to copy a file to one or more directories in which you later want “holes” (empty space) after you have run SPEEDISK. These directories should be placed at the start of the disk, i.e. near its center.

Let’s say that you normally store your data in C:\DOC. Create C:\EXTRA if you don’t have it already. Copy a big file, a half or a whole MB, to this directory. Name or rename the file A.A – for example. Before running a full optimization with SPEEDISK, copy this file to C:\DOC. Now run SPEEDISK and afterwards delete C:\DOC\A.A. Now you have free space in C:\DOC. This place, at the “beginning” of the disk, is where the next file you save will be placed. Of course, you can do this with many directories where you keep data files. Here is a batch file that does this:

```
XCOPY C:\EXTRA\A.A C:\DOC
SPEEDISK
DEL C:\DOC\A.A
```

DEFRAG, which is included with DOS 6.0, can only use conventional memory and lacks the following options:

Directory Order, Full with File reorder, Other Options and even Read after write – Microsoft obviously trusts the program’s integrity.

DEFRAG is not satisfactory for advanced use but even so, it is much better than no optimization at all. It has an undocumented switch

```
C:\>DEFRAG/Q
```

where the Q stands for Quick – and it is *very* quick. Using this method collects all the files in one long row without optimizing every single file, of course. It makes available a block of contiguous space, which can, for instance, be used by a Windows swap file (see p. 42). The H/ switch allows DEFRAG to move hidden files.

Norton Commander (NC)

Introduction

The section differs from the others in that I will walk you through the functions I use the most. So it is a bit like a course in using NC.

This section is here because many people have purchased and use this excellent program, originally written by John Socha. The description mainly relates to the English-language version 3.0.
As time goes by, you cram a lot of files onto your hard disk, and these are spread around in many different directories. I have always used Norton Commander to manage files and directories. If you don't have this program, I'm afraid that you won't benefit much from this section. However, bits here and there might interest you enough to make you consider buying the program. You may well ask (with good reason) why I devote so much space to a program that doesn't come with DOS.

There are many similar programs, including Dosshell, PCTools and File Manager in Windows. I personally find NC the easiest, fastest and smartest.

NC was developed many years ago to ease the management of files and directories. You *can* use a mouse in NC but I will only describe the use of the keyboard.

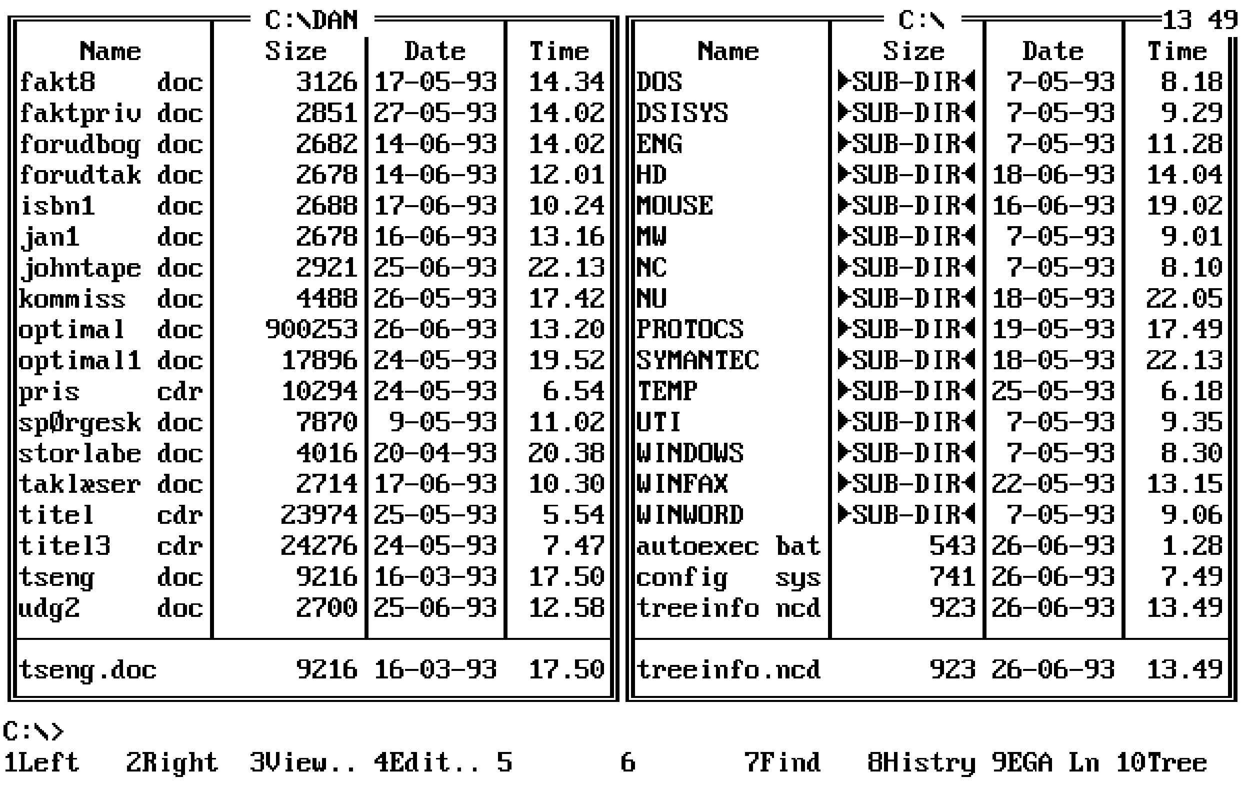
Start NC by typing `NC` at the DOS prompt, and exit by pressing F10. Tab toggles between the two windows. Press Esc if you regret doing something; if you open a window and select a function that you don't want, press Esc. The screen is split into two windows, as if you had two simultaneous DOS prompts, a *very intelligent* and handy function, especially when copying and moving files.

At the top of each window is the paths of the two directories shown by the left and right windows. I call the horizontal marker the "bar." The bar marks a drive, a directory or a file. If the current directory is empty, the bar will be at the top of the screen.

Both windows show files and directories in the manner you stipulate via the pull-down menus. Try pressing F9 and Enter or the down arrow.

This activates the pull-down menus. You can move around in a window by using the up and down keyboard arrows, and you move to a new window using the right or left keyboard arrows. Press Esc twice to return.

The bottom of the screen shows the actions of the different function keys. Try holding down Alt and you will see what Alt + a function key will do.

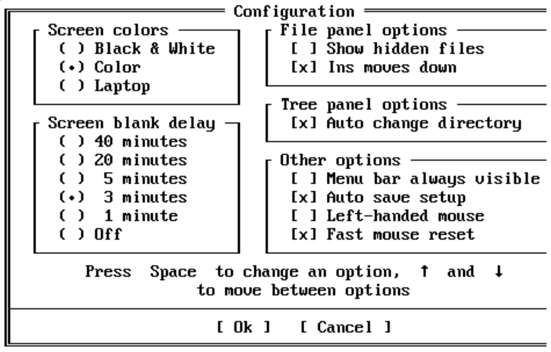


```
                   C:\DAN                                     C:\                 13 49
    Name          Size     Date     Time       Name          Size     Date     Time
fakt8    doc      3126 17-05-93 14.34   DOS         ►SUB-DIR◄  7-05-93  8.18
faktpriv doc      2851 27-05-93 14.02   DSISYS      ►SUB-DIR◄  7-05-93  9.29
forudbog doc      2682 14-06-93 14.02   ENG         ►SUB-DIR◄  7-05-93 11.28
forudtak doc      2678 14-06-93 12.01   HD          ►SUB-DIR◄ 18-06-93 14.04
isbn1    doc      2688 17-06-93 10.24   MOUSE       ►SUB-DIR◄ 16-06-93 19.02
jan1     doc      2678 16-06-93 13.16   MW          ►SUB-DIR◄  7-05-93  9.01
johntape doc      2921 25-06-93 22.13   NC          ►SUB-DIR◄  7-05-93  8.10
kommiss  doc      4488 26-05-93 17.42   NU          ►SUB-DIR◄ 18-05-93 22.05
optimal  doc    900253 26-06-93 13.20   PROTOCS     ►SUB-DIR◄ 19-05-93 17.49
optimal1 doc     17896 24-05-93 19.52   SYMANTEC    ►SUB-DIR◄ 18-05-93 22.13
pris     cdr     10294 24-05-93  6.54   TEMP        ►SUB-DIR◄ 25-05-93  6.18
spØrgesk doc      7870  9-05-93 11.02   UTI         ►SUB-DIR◄  7-05-93  9.35
storlabe doc      4016 20-04-93 20.38   WINDOWS     ►SUB-DIR◄  7-05-93  8.30
taklæser doc      2714 17-06-93 10.30   WINFAX      ►SUB-DIR◄ 22-05-93 13.15
titel    cdr     23974 25-05-93  5.54   WINWORD     ►SUB-DIR◄  7-05-93  9.06
titel3   cdr     24276 24-05-93  7.47   autoexec bat      543 26-06-93  1.28
tseng    doc      9216 16-03-93 17.50   config   sys      741 26-06-93  7.49
udg2     doc      2700 25-06-93 12.58   treeinfo ncd      923 26-06-93 13.49

tseng.doc         9216 16-03-93 17.50   treeinfo.ncd      923 26-06-93 13.49
C:\>
1Left   2Right  3View.. 4Edit.. 5       6       7Find   8Histry 9EGA Ln 10Tree
```

Configuration

Let's start by configuring your screen and program to resemble mine while you read this. You can always change it later. In the *options* menu, select *configuration*. In each dialogue box, you select with the space bar to activate a point and move with the arrow keys, Tab or Enter. Here are my recommendations.

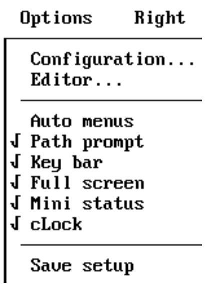
Screen blank delay means that the screen goes (almost) blank after a certain period of time if you haven't touched the keyboard. This protects your screen. Mine is set at three minutes, but you make your own choice.



```
                          Configuration
 Screen colors               File panel options
  ( ) Black & White           [ ] Show hidden files
  (•) Color                   [x] Ins moves down
  ( ) Laptop
                             Tree panel options
 Screen blank delay           [x] Auto change directory
  ( ) 40 minutes
  ( ) 20 minutes             Other options
  ( )  5 minutes              [ ] Menu bar always visible
  (•)  3 minutes              [x] Auto save setup
  ( )  1 minute               [ ] Left-handed mouse
  ( ) Off                     [x] Fast mouse reset

        Press  Space  to change an option,  ↑  and  ↓
                 to move between options

                     [ Ok ]   [ Cancel ]
```

Show hidden files: blank (until you are familiar with NC, I recommend you use a setup that doesn't show hidden files). *Ins Moves down* means that pressing the Insert key moves the bar one step down. *Auto change directory* means that if you have a directory tree in the left window and the corresponding files of a marked directory in the right window, the files of the new directory will be shown each time you change directory in the left window. Under other options, select: *Menu bar always visible* – blank. Menu bar always visible means that the top bar for pull-down menus is always visible. *Auto save setup* – x, means this configuration is saved when you exit NC. End with OK – just keep pressing Enter until you reach OK.

Still under Options (F9 and either arrow-down or Enter). The selections toggle between on and off each time they're chosen. You should have the following active, i.e. showing a small check mark next to them, which you do by pressing Enter or the emphasized letter.



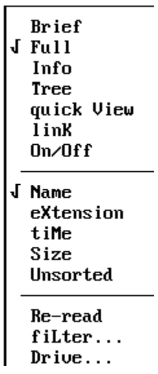
```
Options     Right

 Configuration...
 Editor...

 Auto menus
√ Path prompt
√ Key bar
√ Full screen
√ Mini status
√ cLock

 Save setup
```

Unfortunately, you can only change one thing at a time.

Path prompt – The DOS directory path: The DOS prompt at the bottom of the screen shows the current directory.

Key bar – The function keys are shown at the very bottom of the screen.

F9, and Enter. The menu for the left and right window is split into three sections and within each frame you can choose from one of the following:



```
  Brief
√ Full
  Info
  Tree
  quick View
  linK
  On/Off

√ Name
  eXtension
  tiMe
  Size
  Unsorted

  Re-read
  fiLter...
  Drive...
```

Top Section – Shows the information you choose to see in the window

Brief: file names only.

Full: file names, sizes, dates and times.

Info: information on the current disk, space used and available.

Tree – directory diagram: directory tree in this window and files in the other.

Middle Section – sort order for files

The different options are self-explanatory. Normally, I have *Name* activated but if, for example, you change something in a program that's made up of several files and you want to see in which file the program saves certain things, the most convenient way is to sort by *Time* and the changed file will appear at the top of the list. If you need to view many files with the same extension, then select *Extension*.

Bottom Section – I never use this.

You can select either by moving with the arrow keys and then pressing Enter or by typing the capitalized letter. Try selecting *Brief*; the window shows the current directory with files, though with names only. Do the same with the other window, by pressing Tab, F9 and selecting Brief. If you press F9 and regret it, press Esc. Move the bar up and down with the arrow keys. Notice that at the bottom of the small frame you get full information about the file or directory that the bar is resting upon.. Directories are written in capital letters and files are written with lowercase letters. Now select *Full* for both windows (F9, Enter, Full,), which shows the files with name, size, date and time. This is my preference; you may have a different opinion.

Using Norton Commander!

You must be in the root directory in both windows, so check that it says C:\ at the very top of each window. If it doesn't, then one or both windows are in a sub-directory. Press Home, and the bar will move to *two full stops* (UP-DIR). Press Enter and you will come one step closer to the root directory. Keep going until both windows are in the root directory. Tab toggles between the windows. Move around in the directories and sub-directories so you become familiar with the layout and the controls. Try Home, End, PgUp, PgDn, the arrow keys and Tab.

Place the bar in the right window. We shall make a new directory in the root. Press F7 and call it EXTRA – providing you don't already have a directory with that name. Notice that the bar automatically jumps to the new directory EXTRA. Press Enter. This selects the directory.

At the top of the window, it says C:\EXTRA. The directory is empty and contains no files or sub-directories. Two full stops is a tool that takes you closer to the root when you press Enter. The DOS prompt at the bottom of the screen shows the current drive and directory of the bar. What is very convenient about NC is that while you are manoeuvring around in NC, you can also write normal DOS commands at the normal DOS prompt. Use Tab to toggle between the windows and watch how the DOS prompt changes. Place the bar in the right window so you are in the sub-directory EXTRA.

Take a 3½" floppy disk that contains files. In one corner of the disk, there is a sliding tab. Using your fingernail or a ball-point pen, push it to one side so that the hole is visible. Now the files on the disk cannot be deleted and new ones cannot be written onto it; your disk is now "write protected." You can still read the files, though (the term *reading* a disk means getting information from it, while *writing* mean saving information to the disk). Put the floppy disk in the disk drive.

We now wish to see the contents of the floppy disk, A, in the left window. Press Alt+F1 and press the A key on the keyboard. The screen will be blank for a moment as it reads drive A and shows the contents. Alt+F2 selects the right window in the same way.

Place the bar in the right window. We will now copy all the files in the root directory of the floppy disk to the EXTRA directory.

Copying files with XCOPY

Type (with this prompt):

```
C:\EXTRA>XCOPY A:
```

The files are copied and the left window is updated. With `XCOPY`, you state *what* you want to copy, *from where* and *where to*. Here, we take advantage of the fact that the command inserts standard values for what's not specified. The complete line that we should have written is:

```
C:\EXTRA>XCOPY A:. C:
```

The full stop means "all files." As the prompt is in directory C:\EXTRA, this is inserted automatically for *where to*. The default is all files, so we left out the period/full stop after A:

If we had wanted to copy the other way, that is from EXTRA to A, we would have written

```
C:\EXTRA>XCOPY . A:
```

i.e. "copy all files in the current directory to A:." Most often, you will be copying between two different drives, typically between the hard disk and a floppy disk.

The command `XCOPY` could have been executed without first looking at the contents of the floppy disk with Alt-F1 but I think it is a good idea to look first. You don't even have to be in NC to use `XCOPY`. The advantage of using NC is that you can easily change directories, even at two different locations at the same time.

But make sure you write the right thing, stating the *to* and the *from* correctly. If, for instance, you have all your word-processing files in the same directory from which you regularly make backups, it is important to copy from C to A – and not the other way around because `XCOPY` and `COPY` do *not* warn you (until DOS 6.2) if you are in danger of overwriting files that already exist.

If you are a bit unsure about `XCOPY`, then in the beginning it is best to use NC alone without entering DOS commands at the prompt. Later we'll see how to mark or choose files.

The only drawback of `XCOPY` is that it can't copy hidden files.

Hidden files

Hidden files are hidden for a very good reason: the user shouldn't touch them. They are hidden from `DIR` and `DELete`, which means that you can't delete them in DOS with `DELETE` and you won't be able to see them with `DIR`. This is very reasonable because in normal circumstances, the user does not need to delete a hidden file.

If you want to delete, copy, rename or move a hidden file, you must press F9, options, configuration and put `x` in Show Hidden Files, after which you can manage hidden files with NC.

Selecting files

You will usually select certain files in order to do something with them: copy, delete or move them. Put the bar in the right window. We will delete some of the files that we copied into C:\EXTRA. Make sure you are in C:\EXTRA (it must say C:\EXTRA at the top of the window).

Put the bar on the file and press Ins (on some keyboards, it is called Insert). You can either use the little grey, or the big white, Ins key but with the latter your NUMLOCK light must be off. The whole line will change color, probably to yellow. The file is now selected (marked).

Notice that the bar automatically jumps to the next file. This was determined by the configuration (F9+Options+Configuration) *Ins moves down*. In version 3.0, you can't mark a directory, only files (version 4 can temporarily). Mark some of the files. Move back to one of the yellow (marked) files and press Ins. It will be de-selected. This is how you select files. Select some but not all of them. Press F8 to delete them. First you are warned that you have selected some files to delete. Press Enter. You are warned again and can still back out using Esc or choosing Cancel. Press Enter. The files are deleted.

The function keys are shown at the bottom of the screen. When you press the Alt key, you see the options available with Alt+ combination. F5, F6 and F8 mainly manage selected files, or the file on which the bar rests. So if any files are selected, the actions will only affect the selected files (no matter where in the directory the bar is). The only condition is that it must be in the window (directory) where you want the command to take effect. If no files are marked, the command will affect the file the bar is marking.

We will now move some files from one directory to another. The right window is still in C:\EXTRA and you have some files left (I hope). Tab to the left window. If you are not on C, press Alt+F1 and then C. Go to the root by pressing Home and Enter until you have `C:\` on the top line of the screen.

Press F7 and call the new directory DELETE. Go into the directory. The left window is now in C:\DELETE and the right window is in C:\EXTRA. Tab to the right window. Select some files with Ins. Now press F6. The program says

```
Rename or move x files to
  C:\DELETE
```

F6 and F5 will by default assume that you want to move, rename or copy to the directory that is in the other window – a time-saver. Press Enter. The selected files are moved to the other directory. The files are no longer present in the directory EXTRA but in DELETE.

You may be wondering why a file can be moved so quickly from one directory to another. Though it is called *move*, the file actually remains in the same physical location on the disk. DOS manages all files in the File Allocation Table (FAT). Here, information is kept that shows a file's physical location and which directory it is located in. If a file is moved, it is only the directory name in the FAT that is changed, and that doesn't take much time.

Now, let's copy all the files in DELETE to EXTRA. Tab so that the bar is in the left window. Press the large grey plus key (+) at the very right of your keyboard. A window on the screen shows

```
  Select the files
  *.*
```

I suggest *.* (meaning *all* files), but you can enter something else, if you want to. Pressing Enter marks all the files in the current directory. Notice the bottom line in the window that shows you the number of files and their size. The grey minus key (–) works the other way around; it deselects.

Press F5 and Enter. The files are now *copied* to EXTRA. I mainly use the plus key to see how many files there are in a directory and how much space they take up. If I want to do something with almost all the files in a directory, I first select them all and then deselect the files I don't want with Ins.

Another useful detail concerning the plus key: if you mark and copy files to a floppy disk *but there isn't enough room on the disk for all of them*, the copying process continues until there is no more space on the disk. The clever part now is that NC has unmarked the files already copied. Just insert a new floppy and continue until all files have been copied.

Back to the example. The right window is still EXTRA and the left window is still DELETE. Place the bar in the right window. Select all files again and press F5 and Enter.

NC registers that all the files about to be copied already exist where you are copying to, and asks: Overwrite, All or Skip. In the English version, you can use the keys O, A or S (Esc does the same as S). Pressing O overwrites the current file, A overwrites all selected files and Esc avoids overwriting the current file. (If you are using a foreign-language version of NC, you might have to use other letters.) Experiment until you understand the system. Be aware that pressing Esc once only works for one file, while A works for all files.

Rename

F6 can do three things – 1) move a file to another directory, 2) give a file a new name within the same directory or 3) rename a directory - a directory is actually just a file, though a very special type of file.

When a file is moved from one directory to another, it looks as though it shifts location but in fact it is just given a new name; remember that a file's full name includes drive + directory + name + extension.

Place the bar on any file in C:\EXTRA or C:\DELETE. We will rename the file OLGA.DOC. Press F6, type `OLGA.DOC` and press Enter – and notice that NC blanks when you type the first character. The file has now got a new name. Now we will rename the file KRISTINA.DOC. Press F6 and you need only write `KRISTINA.*`.

The asterisk at the end means that you want to keep the extension. The file is now named KRISTINA.DOC. Similarly, if you want to call the file KRISTINA.LET, you simply type `*.LET`. Tip: if you want to stop F5 or F6 suggesting a move to the other window, use Ctrl +F1 or Ctrl+F2 to close it. The same combination will open it again.

Now NC knows that you don't want to move the file to another directory. This is particularly useful if you just want to make a slight amendment to the name or want a copy of a file with an almost identical name. The file name will be shown, so first press an arrow key (to show you wish to edit), then you can correct the name. If you type a letter, the name will be deleted. Try it!

Sometimes you want to give a new extension to several files that have the same one. Let's say that you have given all your word-processor documents DOC as the extension – or that the program automatically did. You would now like them to have the extension LET instead. Just be sure that you haven't already got a file with an identical name+extension.

You can select them one by one using Ins or mark them all by using the grey plus key and then typing `*.DOC`. You have now marked all files with the extension DOC. Press F6, type `*.LET` and then press Enter. All the files now have the extension LET.

If you want to make copies of all files with the extension DOC, you should use F5 instead of F6 and, the same as before, write *.LET. This places the files with two different extensions in the same directory. Now delete all the files in the two directories C:\EXTRA and C:\DELETE, and finish by deleting the directories.

View/edit text files

F3 and F4 open text files just as a word processor would. With F3, you can only view, not edit. F4 can edit a text file that is no more than 26,464 bytes long. NC has a small editor. If you want to start a different editor with F4, press F9, *Options*, *Editor*, select *Extern* and type the path for the program you wish to use. I use NC's built-in editor to make minor changes in text files such as AUTOEXEC.BAT, CONFIG.SYS, INI files and BAT files. It is easy and fast. You search within a file using F7.

Ctrl+Y deletes a complete line and Home, End, PgDn and PgUp work as they usually do in most other programs. If you have changed something and want to exit, just press Esc and you will be asked whether you want to save the file or exit without saving.

Searching for files

Sometimes you know that you have a file but you have forgotten where it is on your hard disk. Press Alt+F7, write the name, e.g. MUSIC.DOC, and press Enter. Or you can type just some of the name if you can't remember the full name. `MU*` locates all files that begin with MU (muck, munch, music). The program searches the whole drive and lists all the matching files it finds. Move to the desired file, press Enter and you have selected it.

Creating a new file

Shift+F4. Type the name and press Enter twice.

Recent DOS commands

Utilize Ctrl+E is to recall DOS commands used earlier. With Ctrl+X, you browse forward again. This is the same as the up arrow key and down arrow key in DOSKEY.

Switching window on/off

Ctrl+O temporarily removes NC from the screen. Try it. You would normally use it when you write a DOS command and NC conceals the result. If you know that the next entries are DOS commands, and you want to follow them on the whole screen, you can switch off NC's display temporarily by pressing Ctrl+O. Now you are in DOS and can use DOSKEY if it is active. When you type Ctrl+O again, you must use Ctrl+E to recall commands.

Switching windows

Ctrl+U swaps over the two windows. It is useful when you have chosen directories for the two windows – and then decide you would rather have them the other way around. It is probably a good idea from the beginning to make up your mind where you want A and C when you work with floppy disks. There is a certain logic in using an alphabetical order, and as you read the English program instructions from left to right, you should probably keep A on the left. If you have two floppy disk drives, A and B, do the same. This lessens the chance of making mistakes while copying.

Changing directories

Alt+F10 is used to change directory quickly. When you type a letter, NC finds the first directory starting with that letter. As you type more letters in the directory name, NC gets closer to its target. If two directories have the same initial letters, then use Ctrl+Enter to jump to the next one. You can also move around using the arrow keys. Then press Enter. The window in which the selection bar was placed will now show the selected directory.

The file C:\TREEINFO.NCD contains this tree and if it is deleted, NC will re-create the tree structure, and rewrite the file C:\TREEINFO.NCD next time you press Alt+F10.

If you are deep into the tree structure and want to get to the root quickly, use Alt+F10 and press the left arrow key until you reach the root. Ctrl+< or Ctrl+> does the same.

Leafing through directories

F9, *left* or *right* window, Enter, Tree. Move up and down and the other window automatically shows the files in the chosen directory. Good for fast browsing, to see the number of files in the different directories, etc. And if you don't know the name of the file you are looking for, you might be lucky enough to remember it when you see the name.

Comparing two directories

F9, commands, compare is very practical if you are making a backup copy of a directory on the hard disk to a corresponding directory on a floppy disk. You will be told immediately if there is a difference between the two directories. Files that differ between the two locations are marked and you can go through them to see if you can delete some or if you ought to make a backup copy of them from the hard disk to the floppy.

Practice makes perfect

Having (hopefully) read all this and practiced for a while, you should be a champion at managing files, both on floppy disks and on the hard disk. So here is my advice for arranging your hard disk into directories:

When you install programs, follow their advice. When you make your own directories, make as few SUB-directories as possible. It is better to have more directories in the root.

As time goes by, you will work more and more with NC. You will have more files to manage. Place all your data files in a separate directory, i.e. C:\DATA, with sub-directories for every program you use. You are sure to be using a word processor (put your data files in C:\DATA\DOC), a database (data files in C:\DATA\DB) and a spreadsheet (data files in C:\DATA\SPREAD) and so on.

Using NC lets you quickly check out your documents so it is easy to copy them, individually or together, to diskette.

Menu

The file C:\NC\NC.MNU contains a menu that comes up when you press F2. This text file has some handy uses.

Let's take a look at some of the things that might be in NC.MNU. Use the Space bar and Tab for indenting.

```
D:   Change to DOC
‘ this is a comment
      cd c:\mw\doc

Edit CONFIG.SYS
        C:\BAT\ec.bat
```

The first line in each group, which must be placed on the extreme left, appears in the menu on screen. `D:` at the beginning means that by simply pressing **`D`** you activate the command, which is time-saving if you have a lot of commands.

The next line consists of one or more DOS commands. You can make several NC.MNU in different directories, which become the active menus if you move to these directories. This way you can quickly change between many different directories each with its own menu containing different options.

Version 4.0

This version is an outstanding improvement. Some of the news: Just like with File Manager, you can now do things to a directory together with its sub-directories. In configuration, you can mark *Select Directories*, which means that the grey plus key also marks directories. You can also copy directories together with their sub-directories, which helps when making backups.
The editor (F4) can do a lot of different things, search/replace, block functions (F3 starts block, F3 ends and Shift+F3 cancels the block), etc.

Quick View gives information about the selected directory: number of sub-directories and number and size of all files.

There is a built in manager for ZIP and other compressed files (requires that the compression programs are on your path). NC has its own packing program that produces ZIP-format files. Press enter on a ZIP file and it reveals the names of all the files it contains. They can be treated in all the normal ways (copy, delete, etc.); everything except view! It has its own compression program. Select a number of files and compress to ZIP format by using Alt+F5. If you have LHA and other packing programs on your path, then Alt+F6 will unpack (decompress) files in these programs' formats.

Ctrl+F9 prints a file. Ctrl+F3, F4, F5 and F6 sorts by name, extension, time and size.

Another new feature is a built-in communication program, which is excellent. I use it to send and receive files to and from friends and BBSs.

Something I missed in version 3: After a directory comparison, certain files are marked in a directory. Now you can *invert* the marking with the grey ∗ key and mark the unmarked files – perfect, in other words, for when you want to do something with the files that *aren't* marked.

Version 5.0

The new version does not include anything special, in fact certain things are worse than they were. Version 4.0 is the best.

Making backups

Real men make backups, if not sooner, then later. Hopefully real women make backups too, as at some point most of us experience what should not happen – on rare occasions the hard disk breaks down or you delete a file by mistake and discover it too late.

My best advice is to install a second hard disk and use it as backup. The chances of both disks breaking down at the same time are infinitesimal. I have two identical hard disks, and it only takes a few minutes to back up several hundred MB.

I make backups several times a day, whenever I have done something I would resent having to do again. I use Norton Commander for copying files.

Or buy a ZIP drive, where each optical diskette can hold 100 MB or more. If you cannot afford this, then use diskettes and read the following.

For many years, Microsoft has packaged a program called BACKUP with DOS, which is so impractical to use that most people don't bother. There are other programs for backup and DOS 6 includes a mini version of Norton's Backup.

My hard disk contains primarily program files and perhaps only 20% data files that I have created. As you already have the programs on original diskettes and can reinstall them again if something goes wrong, backup shouldn't be necessary for these. The only things you really need to back up are *data files* and *files that initiate* programs. The latter typically have the extension INI.

It is a good idea to keep programs and data separate in different directories. Make a subdirectory, e.g. C:\DATA\DOC for your WP documents. I've often had to install a program several times. If you know in which directories a program creates and stores its files, it is easy to delete the whole thing and begin again with a fresh installation – though Windows is special.

You have to get used to the fact that making backups and maintaining your hard disk and diskettes takes time. At first I didn't worry too much about it, hoping I would remember file names, hoping the hard disk would never be a problem, hoping I could find my way around my diskettes. I later learned my lesson ...

While writing the text of this guide, I copy to diskette about once an hour, and I do this with everything I do. For safety's sake. I have tried losing several hours' work. It is no joke.

I have set up directories on my diskettes that correspond to those on my hard disk so I am in no doubt where the files come from. I usually use Norton Commander as I typically work with several files at the same time but I also use PKZIP, a file-compression program. If you work with large files, it is a good idea to use a program that can pack the files in such a way that they don't take up too much space.

Making backups is very much an aspect of file management. I strongly recommend that you regularly spend some time going through your hard disk and your diskettes. Are there programs or data files on your hard disk that you seldom use? We often install a program to see what it can do and then forget about it. If you have installation diskettes for the program, delete it from the hard disk or pack the relevant directories with a compression program. The fewer files you have on your hard disk, the faster your PC works.

If your files are small enough to enable you to store them in directories that are no more than 1.44 MB, you can easily make a diskette for each directory. With a compression program (like PKZIP, ARJ or LHA) you may still be able to have a diskette for each directory, even though the directory is larger than 1.44 MB. A compression program can typically reduce files sizes by 50%.

I would also generally recommend that if you experiment with CONFIG.SYS, you make running copies of C:\CONFIG.SYS to C:\DOS.

```
C:\>copy config.sys c:\dos
```

so you can always copy it back to the root directory, possibly if you find you have to boot from diskette, as described on page 56.

Windows

Windows is a huge subject but here is a short piece that is relevant to the rest of the text.

I use Windows a lot, and like most people usually have both DOS and Windows programs running at the same time. You should not expect just to be able to install Windows, and then have everything work as fast as you are used from your DOS programs. Updating a screen in *graphics mode* takes considerably longer than updating a screen in *text mode*.

Windows 3.x is *not* an operating *system*. It is an operating *environment*, an *extension* to DOS, a graphic user interface, a more practical, more visual method of managing programs, files, directories, and so on than the traditional DOS command line.

Windows is especially relevant if you want to work with graphics – layout, drawing programs and so on – or if you want a quick impression of *roughly* how something will look when it is printed. You can use the mouse in all Windows programs, and you can click your way through nearly everything. Another convenience is that programs are similarly laid out, similar menu commands and so on (like the Macintosh). Furthermore, you don't have to know or remember many DOS commands.

You can run your usual DOS programs in the manner you are accustomed, by using the full screen. But, if you prefer, they can run in a window smaller than the whole screen. With some programs, you can move data from one window to the other. You can run several programs at the same time, and easily move from one to the other.

The problem with Windows (if you are used to the speed of DOS) is that it needs a fast CPU and loads of RAM. A 386DX/33 MHz with 4 MB RAM is the minimum for acceptable speed.

Windows needs XMS (*extended*) memory and appreciates a fast hard disk and graphic card. By default (assuming sufficient memory), Windows starts in *386 enhanced mode*. Windows' *standard mode* – `WIN/S` – is about 10-20% faster. You could use this when you are only running Windows programs that require smaller amounts of memory.

Swap file

When there isn't sufficient memory available (because you've got more programs open than your RAM can hold), Windows uses the hard disk as extra memory, *virtual memory*. When you shift programs using Alt+Tab, everything that couldn't be held in RAM is summoned from the *swap file*. This is a material improvement in Windows speed – banishing for ever those "out of memory" messages, provided you make your swap file large enough.

You can make your swap file *temporary* or *permanent*. The temporary one has to be set up every time Windows starts, which takes time and can only be recommended if you are short of space on your hard disk. If this is the case, then you would be better off cleaning/tidying up your hard disk so that you can find the space for a permanent swap file, which is much better.
The permanent swap file *reserves permanent space* on your hard disk. Before making this file, run a disk-optimization program or type

```
C:\>DEFRAG/Q
```

To set up a swap file in Windows: Select Main, Control Panel, 386 Enhanced, Virtual Memory, Change. How big should it be? A general rule is that your free XMS memory after booting (but including the swap file) should be equal to 12 MB. 4-8 MB is a suitable size for most people.

You can check, by watching the hard-disk lamp every time you use Alt+Tab, to see if the CPU has to access the swap file to fetch data. If, when you have many programs open and are moving between them, you can hear lots of hard-disk activity as information gets swapped to and fro, try increasing the size of the swap file. It can set at only a certain proportion of the available space on the hard disk. I would also recommend that you activate 32-bit disk access.

32-bit disk access.

This gives faster communication to your hard disk by bypassing DOS and the slow BIOS when swapping to disk. DOS programs will also run faster in enhanced mode. If Windows tests your hard-disk controller and finds it compatible (conforming with) a certain standard (**W**estern **D**igital 1003), then you are able to activate 32-bit access – put a cross in the check box. For technical reasons related to the way that portable PCs save on battery use, Microsoft has *not* set this as the default.

(A) It is unfortunate that Microsoft has chosen to call this communication method "32 bit." Another name is FastDisk. It has nothing to do with the I/O bus or the CPU's address bus width.

It is something technical that works with the 386's address register.

Windows uses a device driver that in *protected mode* communicates directly with the hard-disk controller, increasing throughput by approximately 20% and allowing more DOS programs to be run at the same time.

If you can't start Windows after activating 32 bit disk access, start it with

```
WIN/D:F
```

and turn the 32-bit disk access off.

Win 3.1 can run in *protected* or *enhanced* mode. While Windows is running in enhanced mode, every DOS program is given memory as if it were running on an 8086-based PC. If you have four DOS programs running, you are simulating four of the classic PCs. This *mode* is called Virtual 8086, shortened to V86 mode, and here the 386 processor simulates an 8086 processor, *while* it runs in protected mode.

The advantage is that you can run real-mode DOS programs with the advantages of protected mode, i.e. protection against memory conflicts. Furthermore, you appear to be running more programs at the same time. It looks that way, even though the 386 processor is in fact just shifting rapidly between the different programs, each of which have control of the CPU for a short, precise time. This is what is called *multi-tasking*. Every DOS program also has at its disposal all the available conventional memory, and this is why memory optimization is important.

In principle, a 486 processor behaves in the same way as a 386 here.

Miscellaneous tips for Windows

If you want to save Program Managers settings *without* quitting Windows, hold Shift down while you "exit" Windows – using Alt+F4, for example. Your settings are saved but Windows *does not* close. You can then switch off *Save settings on exit* from the Options menu.

I find it difficult to read the green words in Windows Help. In WIN.INI under `[Windows Help]`, try writing

```
Jumpcolor=0 0 128
Popupcolor=128 0 0
```

where the numbers give red, green and blue values. You can play around a little and see what suits you best. Thanks to Brian Livingston, who passed on this tip: Insert the following in SYSTEM.INI:

```
[386enh]
MaxBPs=768
```

This has solved a lot of problems for many people. It specifies the *maximum number of breakpoints*. A breakpoint is 10 bytes that Windows uses to control DOS sessions. These are DOS programs that each run on their own *virtual* PC, which means that each program behaves as though it is alone on its own machine. To be even more accurate, a breakpoint is used by Windows every time it needs to communicate in *real mode*. To sum up the reason for this command: when Windows starts, it sets aside a certain number of breakpoints by default. When specifying this number, the programmers assumed that it would be more than enough. Unfortunately, this has proved not to be the case, and a Windows session can easily use more than the default number, leading to some rather unpleasant problems. Since I have added this line, I have had fewer program crashes.

As we all know, neither Windows 3.1 or Windows 3.11 is perfect, and both are prone to either lock up or crash at regular intervals. I have got into the habit of exiting from Windows and restarting it, or even rebooting the computer, about once every hour to flush out the memory. Many Windows programs slowly eat your memory up every time they are opened or closed (called memory leakage – programs written in Visual Basic are especially prone to this). I would rather use a couple of minutes every hour to reboot in a controlled fashion than suffer unexpected crashes that might well lose my data. If your language version of Windows produces a comma when you press the period/full stop on your numeric keyboard, you can change it to a period/full stop using a text editor. For example, Danes would change the file WINDOWS\SYSTEM\KBDDA.DLL. Search for `,,**--++` and change to `..**--++` and that does the job. But the usual warning: *before doing this make a copy of the file*. Just in case. Your national keyboard driver has a similar filename.

Other ways to start Windows: type `WIN/?`.

DOS 6

The following description concerns Microsoft's edition of MS-DOS 6. The first thing to say is that simply installing DOS 6 will not make your PC run faster than it did using DOS 5.

I will confine myself to the parts of DOS 6 that are relevant to the rest of the booklet. Accessory programs like Undelete, Anti-virus, Interlink, Backup, and so on are not described, mainly because I do not use them.

Installation

It is possible to install so you can return to an earlier version. I consider this to be unrealistic and so I recommend that you type

```
A:\>setup/G
```

which doesn't need *uninstall diskettes*. If you want to see all the setup options, type

```
A:\>setup/?
```

Be prepared for the installation process to take some time. Certain resident disk-cache programs (not including SMARTDRV), delete-protection and anti-virus utilities that are not compatible with SETUP. I recommend that you suspend these during the installation process by writing `REM` at the beginning of each relevant line in your startup files, and then rebooting with SETUP disk # 1 in the drive.

You have the option of installing (1) only DOS, (2) only Windows or (3) both DOS and Windows versions of the three accessory programs. If you don't install everything now, you can always do so later. Just run Setup again.

I opted for the possibility of reverting to DOS 5, and a directory C:\OLD_DOS.1 was created, to which all files belonging to earlier DOS versions were copied. Note that files in C:\DOS that aren't overwritten by files from DOS 6 with the same name are left in C:\DOS. I would suggest that you make sure that before installation you only have DOS files and nothing else in this directory so that it is easier to find files after the process is finished – in fact I copied the whole directory to C:\DOS5.

I was a little confused when I compared C:\OLD_DOS.1 with the new C:\DOS after installation. I thought that setup would actually copy everything in C:\DOS to C:\OLD_DOS.1 but it didn't. A file in C:\DOS that didn't belong to DOS 5 remained in C:\DOS. Setup works this way so that you can go back to DOS 5, but very few of you will want this.

When you've decided to keep DOS 6, you can delete C:\OLD_DOS.1; for example, by typing

```
C:\>DELOLDOS
```

The startup files are kept in the root directory. Boot with the new DOS 6 and you will see the first changes immediately. For two seconds, the screen shows

```
   Starting MS-DOS..
```

Many users either didn't need, or were confused by, the technical messages produced by programs such as EMM386 and SMARTDRV so they are now removed by default. Good idea. If you want to see this information, as you could with DOS 5, insert `/V` somewhere in the line that calls EMM386 and SMARTDRV.

I was interested to see if anything had happened to my start files. Setup hadn't told me of anything during installation. I was rather surprised to see that now

```
device=C:\DOS\setver.exe
shell=C:\DOS\command.com C:\DOS\ /p
```

had been added, without asking me, and had stolen some KB of my conventional memory. At least SETVER could have been placed in upper memory.

Help

Big changes in the Help function. The whole DOS command manual is now online. For example you can type `DIR/?` to get quick, concise help about all the parameters and switches – on screen. Type `HELP DIR`, and EDIT starts – use the Tab key, or the initial letters, to move around the highlighted topics. It is very useful (and ecologically friendly) to be able to look things up immediately here instead of having to wade through a large book.

A long-awaited command is `DELTREE`, which can delete a directory together with its subdirectories.

```
C:\>deltree c:\extra
```

EMM386.EXE

Using the parameter `RAM` EMM386 now takes all the extended memory under its wing and gives a program what it needs, whether this be EMS or XMS memory. This requires the use of a page frame, which occupies 64 KB in upper memory.

With DOS 5, only a certain amount of XMS could be converted to EMS memory – whereupon it was “locked” in position as EMS. To release it, you had to alter your CONFIG.SYS and reboot your PC.

If you type the following in CONFIG.SYS

```
..emm386.exe ram min=0
```

you will see a screen message during boot to the effect that EMM386 can simulate any EMS or XMS memory a program needs. The above line is recommended if you *sometimes* need EMS memory and if you have 64 KB free in upper memory.

`min=0` means that 256 KB is not reserved from the start, as it would be if `min=0` was not specified.

You will be using 64 K in upper memory (which is taken from extended memory) but you don’t need to concern yourself with whether your programs use EMS or extended memory.

See also page 47 (*Multiple Boots*) for how to get more control over how your memory is allocated.

MEMMAKER

MEMMAKER can do a lot to optimize your startup files.

Start by typing

```
  C:\>MEM/?
```

to see what this program has to offer. The screen information from `MEM` is easier to read and understand in this version of the program. Type

```
  C:\>MEM/C/P
```

and make a note of the information about free memory. That way it will be easier to follow the process when you run MEMMAKER.

Don’t be nervous about running MEMMAKER. You can always go back to the previous configuration of your startup files, *but only one step back*. If you want to go back further, then you can add 1, 2 and so on to the startup files with the extension UMB. Remember that the files are in C:\DOS – not too clever. At the DOS prompt enter this:

```
C:\>MEMMAKER
```

You can run either a configuration option called *express* or another called *custom*. The first thing MEMMAKER does is to copy your existing startup files in C:\DOS, adding the extension UMB to them. It then changes only your startup files, nothing else.

If you want to be able to revert to the previous file, i.e. one generation earlier, type

```
C:\>MEMMAKER/UNDO
```

MEMMAKER inserts certain lines and adds all sorts of parameters and switches to others – and there are a lot of changes!

I tried *express* first to see what would happen. The program booted my PC and ran various tests; this took a while. Screen messages keep you informed of how far MEMMAKER has progressed. Afterwards you can see the differences in your startup files.

The great thing with MEMMAKER is that you can run it again and again until you find the optimum configuration. Keep an eye on the screen in case of problems. I have not experienced any but if you do, you’ll have to look in the manual. If everything gets in a mess, use

```
C:\>MEMMAKER/UNDO
```

A selection is changed with the spacebar, and the arrow keys are used to move up and down between options.

Need expanded memory?

This affects the line with EMM386.EXE and its parameters. Both express and custom installations ask you if any of your programs need EMS memory. If you don’t know, MEMMAKER advises you to answer No. If you answer Yes, `RAM` is inserted in this line, and if No, `NOEMS` is inserted.

These are the two possibilities available for making use of upper memory. As we saw earlier, the `RAM` parameter uses 64 KB for a page frame – so it is important to have this space “spare” in upper memory; otherwise 64 KB will be taken from conventional memory.

Custom

The advanced choice is called both *Custom* and *Advanced Options*. The manual has only a very brief description of the advanced options, and the Help function isn’t much better. The way I understand the information in Help is given below. Remember not to press Enter until *after* you have made *all* your choices. MEMMAKER adds various parameters to the line with EMM386.EXE.

`Specify ..`

If you haven’t had problems while MEMMAKER is running, answer No. If you *have* experienced problems, the cause may be one of the programs loaded when the startup files are read. Answer Yes, and this option gives you the chance to answer Yes/No before every program is loaded and so find out which one is giving the problems.

When you've found the problem program, add `REM` in front of the line loading it, run MEMMAKER, then delete the `REM` so that the program loads. Microsoft recommends that you write the program's name (with or without an asterisk preceding it) in the C:\DOS\MEMMAKER.INF file, which has the same effect as my suggestion.

Scan the upper ..

By default, MEMMAKER (via EMM386) tries to manage all free space in upper memory. If you answer `Yes` to this (and you should answer `Yes` if you haven't had problems here), `HIGHSCAN` is inserted in the EMM386 line. You *can* answer `No`, which means that a safer area in upper memory is scanned. First try `Yes`, and if that doesn't work, use `No`. DOS 6.2 defaults to not using `HIGHSCAN`, i.e. it scans upper memory less aggressively.

Move Extended BIOS ..

Answer `Yes` and get an additional 1 KB conventional memory free. If you run into problems later, then run MEMMAKER again, answer `No` here and see if it fixes the problem. We're really into insignificant detail here.

Monochrome region..

See the drawing on upper memory on p. 14. Many people these days have a "Super VGA" monitor, which displays at 800 x 600 resolution. If you know how to edit Windows' SYSTEM.INI, you should answer `Yes`. MEMMAKER inserts the parameter

```
I=B000-B7FF
```

in the line for EMM386.EXE.

Before running Windows, you edit Windows' SYSTEM.INI and write in the section [386Enh]:

```
device=C:\DOS\monoumb.386
```

If you run EGA or VGA (i.e. with a maximum resolution of 640 x 480), then experiment with `Yes` and see how it goes. If it works, you've "won" 32 KB in upper memory that you can use to run more programs. The acid test is trying to start Windows!

Keep current EMM386 ..

Here you can select what to do with the parameters `I=..` and `X=..`, which include or exclude areas in upper memory. If you really understand the function of these parameters and what they include and exclude, then answer `Yes`, which means that your `I=..` and `X=..` will be kept; otherwise answer `No`.

If you answer `No`, you are letting MEMMAKER do the job, and maybe it can find something better.

Optimize..for Windows

This only affects the translation buffers, data in memory that Windows needs to be able to run DOS programs. At first I thought that this command could do more and actually optimize the whole Windows environment.

If I answer `Yes`, the lines `WIN=EA00-ECFF` and `WIN=ED00-EFFF` are inserted on my PC. MEMMAKER adds all these parameters in the EMM386.EXE line in order to reserve these addresses in upper memory for use as translation buffers, preventing anything else from using these UMBs.

My suggestion is to answer `No` and use `MEM` to check that you have 8 KB free (or 24 KB if you are on a network) in upper memory before Windows starts.

The manual and Help function go round and round the subject of DOS programs, running under Windows or not, and I'm still of the opinion that the manual is hopeless on this point. If you set up a CONFIG.SYS with menus (as described on page 47 *Multiple Boots*) and then run MEMMAKER, MEMMAKER cannot understand what to do. The only way around it is to divide them up into separate start files, run MEMMAKER for each of them and then merge them together. Very complicated!

Important: Memmaker *cannot* change the order of devicehigh, which is the most important aspect of memory management. You have to do it manually.

In general, I would say that Memmaker is better than nothing, but I am not impressed.

More booting

F5

Try booting and pressing F5 as soon as the `Starting MS-DOS..` appears on the screen.

This now produces what is called a “clean boot,” which means that your startup files are not read. If you type `SET` at the command prompt, you can see that the PC has had a nearly clean boot. F5 is great if you experiment with CONFIG.SYS. You don’t need a boot diskette, if the problem is simply a mistake in CONFIG.SYS.

This is also useful before running a disk optimization program or something similar.

DOS 6.2: If you have installed Doublespace, you can bypass it by pressing Ctrl+F5. The point of this is to solve any problems that may stem from DBLSPACE.

F8 and ?

Boot again, and this time press F8. Now you get the chance to answer `Y` or `N` to every single line in CONFIG.SYS and after that as to whether you will run all of AUTOEXEC.BAT or not. In DOS 6.2, you can type `Y/N` to every line in AUTOEXEC.BAT.

If a question mark is inserted immediately after a command in CONFIG.SYS (e.g. NUMLOCK?=ON), you will be asked if you want the command carried out.

DOS 6.2: Ctrl+F8 bypasses DBLSPACE in the same way as Ctrl+F5.

Multiple Boots

Users who need several different configurations or share a PC with other users find life easier now. You can make your configuration and boot more sophisticated with menus in CONFIG.SYS, allowing you to select between different options (shown in the frame).

Text after the comma in the `menuitem` line is shown on screen during booting. The first lines are the block headers that relate to the names of the blocks following. These names must be identical!

You can write what you want but you’re limited to a single word as a block name. I suggest you insert a `[COMMON]` line as the last line in CONFIG.SYS to deal with programs that add one or more lines to CONFIG.SYS during installation. DBLSPACE does this.

Even an empty block with `[COMMON]` does no harm. There can also be lines that are common before the `[menu]` entry. I have experimented to get the maximum amount of conventional memory for games that need it. If you have a better idea, I’d like to hear from you. Some games need conventional memory. Others can use expanded memory. The variable `%config%` obtains its value from your menu choice, and then jumps to a “label” i.e. `GAMES1`.

I get 625 KB free in conventional memory, and 622 KB if I also activate expanded memory. In AUTOEXEC.BAT, I have also specified SMARTDRV for `GAMES1`, which gives access to upper memory, so you can check if your game runs faster using SMARTDRV.

If you only have DOS 5 but are reading this section about DOS 6 anyway, no need to despair. Write three versions of your startup files that correspond to the examples above. Place them in your DOS directory, and give both files in each set (CONFIG.SYS and AUTOEXEC.BAT) the extensions NOR, GM1 and GM2, which then correspond to the following BAT files:

```
C:\BAT>COPY CON NORMAL.BAT
COPY C:\DOS\CONFIG.NOR C:\*.SYS
COPY C:\DOS\AUTOEXEC.NOR C:\*.BAT
```

Press F6, Enter. You have created NORMAL.BAT. By the same method, create two more batch files called GAMES1.BAT and GAMES2.BAT. When you want a “normal” boot, at the DOS prompt write:

```
C:\>NORMAL
```

which copies your normal startup files to C:\; then reboot the machine. In the same way, when you want to play a game, either with or without expanded memory, use your GAMES1 and GAMES2 batch files. You may also create a batch file for every game and place these in, for example, a directory called C:\BAT, or any other directory that lies on your PATH. I hope that this description hasn’t been too short.

NumLock

`NUMLOCK=ON` or `OFF` in CONFIG.SYS means....? You guessed it!

```
DEVICE=C:\..\HIMEM.SYS
DOS=HIGH

[menu]
menuitem=GAMES1, with expanded memory
menuitem=GAMES2, conventional memory
menuitem=NORMAL
[GAMES1]
DEVICE=C:\..\EMM386.EXE ram min=0
DOS=UMB
STACKS=0,0
FCBS=1
BUFFERS=10
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /P

[GAMES2]
STACKS=0,0
FCBS=1
BUFFERS=30
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /P

[NORMAL]
DEVICE=C:\..\EMM386.EXE NOEMS
…and so on.
[COMMON]
```

You might need `FILES=15` or more. In `GAMES2`, I use 30 buffers. As mentioned before, MEMMAKER can't be used to optimize this sort of CONFIG.SYS, as it can't cope with an `[INCLUDE]` setting.

AUTOEXEC.BAT looks like this:

```
PATH C:\DOS;C:\BAT;C:\;C:\WINDOWS
goto %config%
:NORMAL
@echo off
…and so on with normal AUTOEXEC.BAT
goto end
:GAMES1
LH smartdrv
goto end
:GAMES2
:end
```

MS-DOS 6.2

The most important reason for releasing DOS 6.2, according to Microsoft, was to improve DBLSPACE, compared to version 6.0, which is why Double-Space is in this section. But first, a little on the other improvements:

`SCANDISK` is a program that can investigate and repair hard-disk errors in a similar way to Norton's Disk Doctor. Microsoft intends it to be a replacement for CHKDSK. The program can also test and repair errors on a drive compressed by `DBLSPACE`.

`VERIFY=ON` in AUTOEXEC.BAT can cause problems with certain hard disks (Conner). Therefore, delete this line, if it exists.

The `DISKCOPY` command, which makes an identical copy of a diskette, can at last carry out this operation by only needing the original disk in the drive once, followed by the copy disk; then the process is complete.

`COPY, XCOPY` and `MOVE` now warn if you are about to *overwrite* a file.

`DEFRAG` can now use extended memory and so can manage a larger number of files.

HIMEM.SYS now supports 64 MB RAM and by default checks the RAM chips in extended memory during booting.

Disk compression with DBLSPACE

In recent years, the need for more hard disk space has grown enormously. In 1993, at a time when hard disks were still relatively expensive, many firms developed a variety of programs that packed (compressed) data on the hard disk so that it took up less space than it otherwise would. And then the price of hard disks fell drastically! If you can afford to buy another hard disk, then forget all about this section on DoubleSpace. You can always use your old, slow hard disk for making backups.

Microsoft included a compression program called DBLSPACE with DOS 6.0 and 6.2.

When data is used, it is unpacked, and when a program stores data, it is compressed. The aim is to prevent a drop in program speed, so that the user doesn't notice the unpacking process.

Use Dblspace?

DBLSPACE certainly caught the public imagination. Some people were certain that the DOS 6.0 version was buggy, which Microsoft has perhaps *indirectly* confirmed by releasing DOS 6.2.

I have tested this program briefly so I can comment on it, and I have not experienced any problems with it.

Will DBLSPACE slow down my system? This is what Microsoft has to say in HELP:

"If you have a computer with a fast CPU and a fast hard disk, you probably won't notice much difference in system speed after installing DBLSPACE. If you have a fast CPU and a slow hard disk, DBLSPACE might actually improve your system's speed. If your computer has a slow CPU, you may notice a reduction in speed after compressing your drive."

Reasonable enough, but what is a fast CPU, etc? My guess is that a fast CPU was at that time at least a 386DX/40 MHz, a slow hard disk is a standard hard disk bought before 1992 (with a transfer rate of 5-700 KB per second), and a slow CPU is a 386SX. To check how correct this is, you will just have to try it out!

If you are short of space on your hard disk, then begin with the kind of installation suggested here and see how it works. Remember that you will need 38 KB of upper memory free if you want to avoid using any of your conventional memory!

If you decide to use this program, then I would suggest that before installation you:

1 Upgrade to DOS 6.2 because this version of DBLSPACE has been improved and is more reliable.
2 Back up all vital files.
3 Defragment your disk. Choose the `FULL` option to create the maximum possible free space.
4 If you are on a network confirm that you are logged on *before* you run DBLSPACE.

I assume that your hard disk consists of only one drive, the C: drive, so when I write about a *drive* in these instructions, this is equivalent to a *disk*. If you have DOS 6.2, then SCANDISK will start automatically before the actual disk compression begins.

If you want to know more, then you can read the manual or write `Help Dblspace` for extra information before beginning the process.

There are two ways to install DBLSPACE.

```
C:\>Dblspace
```

The first screen informs you that Setup is loading the file DBLSPACE.BIN, a system file that needs about 40 KB in memory. The next screen gives a choice between the two possible options, *express* and *custom*, and explains that express is the easiest, while custom is for the more experienced user. This is correct but what Microsoft *does not* explain is that it is much harder to change your mind and uninstall an *express* installation, which is a compression of your *entire* drive C:.

Using *express* compresses a whole drive.

Custom creates a new drive on your existing drive, using the free disk space.

Most users will probably choose to compress their hard disks by using the *express* option, which is the simplest, but also the most difficult to change, i.e. *undo*. In order to gain some experience, or if you are a little unsure, I would suggest that you start with *custom*, which is easy to remove again.

The new drive, no matter which method you choose, must have a letter as a drive denominator. DBLSPACE jumps four drive letters from the last existing drive, so if you start with drive C:, your new drive will be called H. This is so that there is room for other programs (a ramdisk or netdrive) that wish to use a different drive letter from the drives already physically installed.

Your new drive H is different according to which method of compression you use, *custom* or *express*, but I will return to that.

Custom

I suggest that you start by creating a small DBLSPACE drive to gain a little experience, and especially to test whether your PC slows down. If you decide later to drop DBLSPACE, you can do it easily and return to your normal configuration. If you decide later that you wish to compress your *entire* hard disk, you can change your *custom* installation reasonably easily, and by then you will know more about DBLSPACE. So, with this in mind, choose *custom*.

The choices on the next screen are confusing. You are presented with two options:

Compress an existing drive
Create a new empty compressed drive

(If you only have one drive – drive C – then the first option is the same as *express*, and we have just told the program that we do not want an *express* installation. DBLSPACE should be able to test how many drives the machine has.)

I suggest that you use Enter to choose the Create a new… option. This results in the next screen presenting you with a figure for how much space you have available (current free space) and an estimate of how much free space a compressed drive will give you (projected size of new drive). Make a note of the first figure; you will need it later.

Press Enter and the next screen will tell you that the new compressed drive H will be created from the *free* space on C. Now comes the important part. The top line in the frame specifies that a miserly 2 MB will be left on drive C after compression (note that this is free space; it has nothing to do with your files in C). If you started with 50 MB free space, then 2 MB will remain in uncompressed form. DBLSPACE will compress the other 48 MB physical free space so that it can contain about 96 MB of files.

I suggest that you alter this 2 MB to a larger figure, so that you end up with a compressed drive that can contain about 20 MB. If you had 50 MB free, then use the arrow key to move up to the 2 MB and press Enter. In the next screen, write 40 MB and press Enter again. This gives DBLSPACE (50 - 40) = 10 MB of free space to create the new drive H.

It is a disappointment that the amount of free space is not shown on each screen. DBLSPACE is the big attraction in DOS 6 and one would have thought that Microsoft would have made it more user-friendly!

You can cheat by writing a ridiculously high figure, like 999, and the program will then tell you what the maximum figure can be, which is the amount of disk space free. The programmers have measured the free space, but they cannot be bothered to tell the user what it is.

The last screen before the process starts tells you that DBLSPACE is ready to begin, and you are requested to press C to continue. When I tested the program, it estimated it would take 53 minutes; in fact it took 10. SCANDISK starts first and if it runs into any problems (*lost clusters* or *crosslinked files*), you will have to abort the installation and fix these (see CHKDSK, p. 55). If you do this using SCANDISK, it is not necessary to perform a *surface scan* at the same time, as DBLSPACE will carry one out later. After fixing any problems you can start DBLSPACE again.

(A) SCANDISK now carries out a *surface scan*, and if this reports that all is OK, the PC is rebooted and the file C:\DBLSPACE.INI is created. DBLSPACE.BIN is copied from C:\DOS to C:\ and both files are given system, hidden and read-only attributes.

The last screen gives the new compressed drive's size and the amount of remaining (uncompressed) space left on C:. Note that drive C: is still drive C: but is now host to (contains) the new, compressed, drive H:.

Express

If you choose this, *easier*, method, then nothing appears to change except that your hard drive gets bigger, and maybe slower. You can use your PC as you always did, but if problems arise or you decide it was not a good idea to compress your disk, it might help if you understand what is going on.

(A) DBLSPACE creates a drive, typically H, called the *Host Drive*. (It is just a coincidence that H is the initial letter of *Host*.)

In an Express installation, this drive H: is *not* compressed, and actually contains all the data from C: in a large compressed file known as a *Compressed Volume File*. Note that the Express installation is the reverse of the Custom installation, and here the H: drive is host to drive C:, which exists as a file called DBLSPACE.000 and is “converted” to drive C: during booting (before CONFIG.SYS is loaded). This is because DBLSPACE.BIN is part of the DOS 6 operating system and is a system file no less important than the other two. If you create several compressed drives using DBLSPACE, they will have sequential extensions, i.e. DBLSPACE.001, etc.

The other system files and boot files are placed on the H: drive, along with hidden files from other programs and your Windows swap file (see below). These files cannot function in compressed form.

Warning: *Never touch/delete/move the files belonging to DBLSPACE; you risk losing all your other files at the same time.*

A Windows Permanent swap file can only exist on an uncompressed drive. If, before installing DBLSPACE you had a permanent swap file, it should have been moved to H: during the installation process.

If you *did not* have a permanent swap file (perhaps you did not have Windows) and later you decide you need one, you might find – although I have not personally tried this – that you need to create more free space on drive H. You can do this by starting DBLSPACE and using the Resize option in the Drive menu to increase the free space on drive H: so there is enough for your swap file. You end up with the same result by giving the command:

```
C:\>DBLSPACE/SIZE/REServe=8
```

where `reserve=8` means: reserve 8 MB on the uncompressed host drive.

Custom and express

This line is added at the end of CONFIG.SYS:

```
devicehigh=C:\dos\dblspace.sys/move
```

Move the line to make it the first devicehigh line. This ensures that DBLSPACE.BIN is read into upper memory.

(A) DBLSPACE.SYS loads DBLSPACE.BIN and if the line in CONFIG.SYS is buffers=8 or less, there will always be room in High Memory for part of DBLSPACE.BIN (see `Buffers=`, page 24).

In general

Data compression explained simply: Imagine a document in which many identical words occur (and, if, when, then, etc.). Instead of having to store the word every time, there could be a very short code for each word. All that needs to be stored is the code plus a pointer indicating where to find the answer. The risk of error if something goes wrong in a compressed file is higher than with “normal” storage. It would be like having the same pin-code for hundreds of different credit cards – and then forgetting the code!

DBLSPACE.BIN is both a “disk space manager” and a program that can compress and decompress files. Since DBLSPACE is an integral part of the operating system, it is loaded *before* CONFIG.SYS. DBLSPACE.BIN is first loaded into conventional memory, but later (via the line added in CONFIG.SYS) it is placed in upper memory.

The improved version of SMARTDRV, included with DOS 6, can hold data before DBLSPACE decompresses it, which means it can hold relatively large amounts of data.

Do not use Norton’s Disk Doctor on a DBLSPACE drive - use SCANDISK instead.

Uninstalling

It is easy enough to remove a compressed drive but it requires enough free space on your hard disk to take the uncompressed versions of the files that were compressed. You might find that you have to move some files to disk or tape.

Start DBLSPACE and choose uncompress from the Tools menu. If you only have one compressed drive, you will be asked if you want to remove DBLSPACE completely. Reply YES to this question. (C:\DBLSPACE.INI and C:\DBLSPACE.BIN are deleted, along with the compressed drive DBLSPACE.000. The line containing DBLSPACE.SYS is deleted from CONFIG.SYS.)

Windows for Workgroups 3.11

This section is placed here because the program is more recent than DOS 6.2.

Windows for Workgroups 3.11 is a minor update to Windows 3.1. It can be used on any PC. The main improvement is faster communication with the hard disk by the use of a 32-bit *file-access* system (a disk cache), though in addition the program is reputed to be generally more stable. A small fax program and a new faster print manager are also included.

On installation, a line is added to CONFIG.SYS

```
device=C:\WINDOWS\ifshlp.sys
```

which you should change to device*high*. Win 3.11 includes the same *versions* of these files as DOS 6.2: SMARTDRV.EXE, HIMEM.SYS, EMM386.EXE, but the installation changes your startup files to load them from the Windows directory, which typically is C:\WINDOWS. All very well – but be careful if you run MEMMAKER any time afterwards as this changes the path to C:\DOS. Why does it do that?

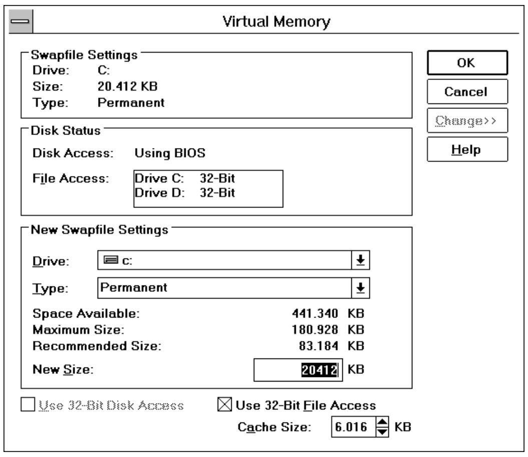
My guess is that when DOS 6 was released, Microsoft wanted to “help” Windows 3.1 users who installed the new DOS to make the “correct” decision, which is to use the newer and better version of the files from C:\DOS. If you *do not* plan to alter your DOS setup, you can *copy* the files from C:\WINDOWS to C:\DOS. Otherwise you will have to change your startup files after running MEMMAKER.

The new 32-bit *file-access* system is in fact a *cache* that replaces SMARTDRV when Windows is loaded. In the dialog box where you specify your swap file (Control Panel, Enhanced, Virtual Memory, Change), you will see the drive that uses the new cache after you mark the 32-bit file-access check box. The size of the cache is also displayed. The program selects a value depending on how much free extended memory you have (same as SMARTDRV does). The installation process alters the Windows cache-size parameter for SMARTDRV (the second figure given after the SMARTDRV command). With 8 MB RAM, it is changed to:

```
..\smartdrv 2048 128 /X
```

One of the differences between SMARTDRV and the new 32-bit file-access program (VCACHE.386) is that SMARTDRV reads some additional data from the *sector* after the most recently read in the hope that the next data required will be in the next (physical) sector on the disk.

The new cache reads its extra data from the *next part of the file* that has just been read, i.e. in the hope that the next data required will be from the same file. As this is highly likely, it gives an improved “hit” rate (and fewer misses) than the old arrangement.


Virtual Memory

Swapfile Settings
Drive: C:
Size: 20.412 KB
Type: Permanent

Disk Status
Disk Access: Using BIOS
File Access: Drive C: 32-Bit
Drive D: 32-Bit

New Swapfile Settings
Drive: c:
Type: Permanent
Space Available: 441.340 KB
Maximum Size: 180.928 KB
Recommended Size: 83.184 KB
New Size: 20412 KB

☐ Use 32-Bit Disk Access
☒ Use 32-Bit File Access
Cache Size: 6.016 KB

(I have not checked 32-bit disk access, as my disk controller runs this automatically.) The new cache will be most effective when the user rarely – if ever – runs a disk-optimization program! If you regularly defragment your hard disk with SPEEDISK, DEFRAG or something similar, there is little or nothing to be gained from using the new method.

Another difference is that the new cache switches some processes from *real* mode to *protected* mode, which makes everything work faster.

If you *always* run DOS programs from Windows, you can delete the `SHARE` line from AUTOEXEC.BAT.

To sum up: in my experience, Win 3.11 speeds up hard-disk access but not much. On the other hand, other functions, such as opening/closing windows and scrolling in dialog boxes, are faster.

Miscellaneous

Optimization tips

You will by now have realized that there is no general way to optimise a PC. We all use our computers in individual ways to suit our personal requirements. My advice is: try out some of the idea suggested in this guide – experiment a little! You will learn something in the process and perhaps be forced to think about things you haven't needed to think about before – and that does no harm.

Generally, it is the case that the less RAM you have, the more important your use of memory becomes. I realize that I have given both minor and major tips, so here are the most important ones:

1) DOS=HIGH
2) Disk cache
3) Optimizing your hard disk
4) Permanent swap file

2 MB RAM
With a 286 and an upper memory manager installed, or a 386SX/SL (on which EMM386 can create UMBs and DOS can manage them), it is important to install as much as possible in UMBs. If you have an *add-on EMS card with RAM*, all you need to do is use that EMS RAM with the driver supplied with the card. Read the manual that came with it and/or software for further instructions.

You must load `DOS=HIGH` and also if possible `DOS=UMB`. It all depends on how much XMS memory you "borrow" for use as UMB. If, for instance, you have about 800 KB RAM left, you have to decide how you will use it. If you only run DOS programs, then use a fair amount (maybe 512 KB) as a disk cache, e.g. SMARTDRV.

The cache size (the 512 KB just suggested) is taken from extended (or expanded) memory and does not affect your conventional memory. If you don't use programs like Windows, you can use all the free extended memory for your disk cache.

If you use Windows, then you have to strike a balance between the size (minimum and maximum values) of SMARTDRV and the free extended memory available to Windows. I haven't tried, but 400 KB each can't be far off the mark – you may want to experiment.

4 MB
You will certainly have at least a 386DX. The text should help you a lot. As I have indicated, SMARTDRV will assume a default value of 1024 KB cache under DOS and 512 KB cache under Windows. These are OK for most users. If you run Windows, try reserving a larger cache for Windows, i.e. try writing the following in AUTOEXEC.BAT:

`smartdrv 1024 768` or
`smartdrv 1024 1024`

If you usually only run one or two less memory demanding programs, then you might well find that they run faster with a cache larger than 512 KB.

6-8 MB
It is not easy to give general advice on the size of the disk cache or the possible use of a ramdisk. It really depends upon how many programs you usually have open in a Windows session, and also *which* programs you use. It would be too easy for me to say you should experiment. Just to give you some idea of my own setup: I have 8 MB and usually have Word for Windows, a database and maybe NC and/or Winfax open. I get the fastest results with SMARTDRV set to 2048 2048, which is the default for 8 MB RAM. In other words, I don't need to state any parameters for SMARTDRV.

With 8 MB RAM, I get more speed during a Windows session with several programs open, and only a minimum of data needs to be sent to the permanent swap file.

Using upper memory

(A) If you want to place as many programs as possible in upper memory, it is important to create a large area with consecutive upper memory blocks (UMB). Do not include more than you need in any `I=...` line in your CONFIG.SYS.

You should load first those programs that take up the most space, and then smaller and smaller programs. The first program loaded is placed in the largest vacant UMB, the next in the largest *remaining* free UMB, and so on. This process leaves gaps of unused memory.

By using the `MEM/C/P` command, you can see fairly accurately how much each program uses. The command `MEM/D/P` also provides useful information. You may have to change the order of your `devicehigh` and `loadhigh` commands.

Common problem: You have found out how much room a program will use, and apparently there is room in upper memory for it – but it will not load there.

Cause: Programs are not loaded directly into upper memory. First, they are loaded into conventional memory, then a check is made for space in upper memory, and only if there *is* space will the program be loaded there. However, during this shunting procedure, programs that are preparing to move to upper memory require more space than they do after they have moved. Afterwards, the room they no longer need in upper memory is freed up again.

In MEMMAKER.STS, *MaxSize* gives the number of bytes the program needs to come "up." If you cannot fit a program in upper memory, then temporarily unload some others, run MEMMAKER and then look in MEMMAKER.STS.

In order to load as many programs as possible into upper memory, the most important thing is *the order of devicehigh and LH commands*. MEMMAKER *cannot* help with the loading order. *You* must decide that.

If you use Windows and do not have any network drivers installed, make sure you have at least 8 K left in upper memory for *translation buffers* after loading all your programs. If you have network drivers installed, this figure must be 24 K.

Windows needs what are called translation buffers when running in enhanced mode, and they fill either 8 or 24 KB. Translation buffers are used to transfer data between real and protected mode - DOS runs in real mode and Windows runs in protected mode, and when a DOS program runs in a DOS window in enhanced mode, then this movement between states is occurring all the time. These buffers are then of importance as temporary storage space for vital information or data.

Windows: make sure that you use your startup files only to load those programs (device drivers, TSR programs, etc.) that *all* your programs need. If you only need a TSR program while you use a certain DOS program, the most efficient policy is to write a batch file for that program that loads the TSR *before* you load the DOS program. When you close the program, the TSR is removed from memory.

Looking in memory

(A) If you have the program MSD.EXE (from Windows 3.1 or DOS 6), you can run it in DOS and type M to see how upper memory is being used. The grey area is reserved for system use. F=Free, U=Used. As already mentioned, the PS/2 has its motherboard BIOS ROM in the area E000-EFFF, so this area is *not* included in the default setting of DOS 5's EMM386.EXE.

However, clone PCs don't normally need this area for system use; therefore, it will be wasted if it is not made available for *upper memory*. This is only important if you need to use part of this 64 KB in upper memory. The parameter `I=E000-EFFF` includes this area. In this way, you can "gain" 64 KB in upper memory, but check first with MSD.EXE that the area is free (shown by Fs).

While in MSD, select Utilities (`Alt+U`) and Memory Block Display. You can look at the first MB, with the possible exception of pages E and F, i.e. from E000 onwards. While Windows or another program is loaded, you can switch to `MSD` and find out where the different programs are located in memory. Notice, however, that (as the screen message says) information may not be totally accurate as you also have Windows running. Despite this reservation, it is useful to check when you want to see if a change has worked.

You can find out a lot about your PC by choosing Alt, File, Print, File and Enter. You can get roughly the same result by typing

```
C:\>MSD/P sysinfo.doc
```

This filename is my suggestion but, as with all DOS commands, you can request info about the different possibilities by typing

```
C:\>MSD/?
```

DOS tips

Deleting all files in a directory

Another reader wrote me (thank you) with a neat suggestion for avoiding those time-wasting *Are you sure?* messages. Honestly, Microsoft can't have a very high opinion of us users and our IQs. Write a batch file called, for example, ERASE.BAT that contains the following:

```
echo y | del *.*
```

When you want to delete *all* files in a directory, at the DOS prompt type: `erase.`

If you have installed a large program that has created many sub-directories, and you want to delete all of these directories and their contents from disk, there is only one way to do so in DOS 5: the long, hard way, starting "backwards" with the "deepest" directory and deleting one sub-directory at a time. Windows File Manager and the DOS 6 command `Deltree` can delete a directory with associated sub-directories with one keystroke.

```
C:\>DELTREE C:\EXTRA
```
and answer Y.

You will often have to delete all files on a diskette. I make it a habit to check exactly what I am about to delete, so I use NC and Alt+F1, look in the files and write ERASE at the prompt to run my batch file. If there are many directories on the diskette, consider quick formatting it by typing

```
C:\>FORMAT/Q A:
```

or an unconditional formatting (you cannot UNFORMAT it afterwards) by typing

```
C:\>FORMAT/Q/U A:
```

CHKDSK /F

I have mentioned this several times in this little guide. Before DOS 6.2, when this program was replaced by SCANDISK, it was used to repair disk errors. If you have DOS 6.2, you can use SCANDISK instead of CHKDSK.

A typical error (from CHKDSK) on a disk is *lost clusters* or *cross-linked files*. Lost clusters are bits of data that do not have a name attached to them, while cross-linked files refer to a condition where two files share, or are linked to, the same place on a disk.

Lost clusters: If it finds any, answer YES to fix them. They will be named in a sequence starting with FILE0000.CHK and will be placed in your root directory. Inspect them and decide whether to keep them (rename the file or files with a more meaningful name) or delete them.

Cross-linked files: If CHKDSK reports cross-linked files make a note of the filenames, copy the files under a new name to somewhere else on the disk, and delete the originals. That also removes the link to the same place on the disk.

From, to

This may not be very relevant for English-speaking readers, but I'm including it because this command gives you the chance to revert to your computer's internal symbol set. If the DEL key on the number pad on your PC produces a comma instead of a period, as it does on many European machines, you can change it by pressing Ctrl+Alt+F1 (change it back by pressing Ctrl+Alt+F2). This toggles between the internal symbol set in your PC and the standard you have set in your startup files.

It works at the command prompt in DOS programs and in a DOS box in Windows. You lose any other country-specific symbols. So you will have to experiment with the £/$ sign and so on to see if you still have them, or to discover where they have been moved to. In any event, you can just swap back and forth between hardware and software symbol sets as you wish.

FDISK

This program, used to partition a hard disk before it is formatted, has an undocumented switch that is quite harmless, but in some circumstances can remove a virus in the master boot record:

```
C:\>FDISK/MBR
```

Boot diskettes

After looking in detail at startup files, it would perhaps be a natural step to discuss a couple of disks you may need if you run into trouble.

A *boot disk* is also called a *system disk*. A *setup disk*, on the other hand, is something different; it is a floppy disk that installs a program – in this case DOS – from the floppy to the hard disk. You may have a setup diskette but you don't necessarily have a boot diskette.

Boot disks can give you a helping hand in times of real trouble. Experience proves it is better to be safe than sorry – especially if you have DOS 5. If you begin experimenting with the contents of CONFIG.SYS, you *must* have a boot disk. Often in my experiments, I just couldn't boot from the hard disk. An error in the CONFIG.SYS file can *stall* your computer, i.e. prevent it from completing the startup procedure. Luckily, DOS 6 has solved this problem.

A corrupt COMMAND.COM file can also cause the PC to crash. I remember once "just" opening the COMMAND.COM with the editor in Norton Commander (I only wanted to take a look) and then closing it without poking around. My computer went on strike. Moral: *never* touch COMMAND.COM.

Unfortunately, a PC can't simply be switched on like an electric toaster and work – it would be great if it could. It must first activate the operating system. Even though you have DOS, and one of the setup disks *can* start your PC, it still won't do the job of a boot disk. So I suggest you make one. A PC can only start (boot) from drive A or C. Format a floppy like this:

```
C:\>FORMAT A:
```

Disk 1 – clean boot

Place an empty, formatted floppy disk in drive A and type:

```
C:\>SYS A:
```

The screen shows `system transferred`, which means that two "hidden" system – or boot – files and COMMAND.COM have been copied to the disk (DBLSPACE.BIN will also be copied, if you have DOS 6). These are the files necessary for DOS to start working, i.e. be read into memory. If you use DBLSPACE, you will also need DBLSPACE.BIN if you want to be able to read files on a disk that it has compressed. Copy SYS.COM to the disk:

```
C:\DOS>COPY  SYS.COM  A:
```

Label the disk *Clean Boot*.

Now you have a boot disk that will always start your PC if it refuses to start from the hard disk. This boot disk only contains the 3-4 files essential to the operating system, and booting from it results in a *clean boot*. Note that this can change the position of certain symbols on your keyboard as no keyboard drivers or codepages have been loaded. The new positions will correspond to those on an American (US) keyboard.

If you want to see which files are on the disk, you can use this undocumented DIR that includes a comma and shows hidden files:

```
A:\>DIR,
```

Some games need a clean boot, so you can use your boot disk. It can also solve two problems you might run into.

Problem 1

You are unable to boot from your hard disk, and get the message `Non-system disk`. If the problem is just that there is something wrong with one of the system files or COMMAND.COM, then boot with your disk, and after booting write

```
A:\>SYS C:
```

which copies the 3-4 files to C:\. When you see the message `system transferred`, you should be able to boot from your hard disk again.

Problem 2

If you get the error message `missing or bad Command Interpreter`, then something has gone wrong with COMMAND.COM. If this happens, boot with your clean boot disk and write:

```
A:\>copy command.com C:\
A:\>copy command.com C:\DOS
```

which copies a working copy of COMMAND.COM from the diskette to the two relevant directories on the hard disk. You might actually need a copy only in a single location but put it in both to start off. Remove the disk from the drive and boot again.

Diskette 2

Take another formatted floppy and do the same as you did with the first one.

```
C:\>SYS A:
```

This diskette will be used to boot your machine in the same way as a boot from the hard disk – except that it will be done by the floppy. All the essential files should be on the floppy (here's the advantage of having startup files without C: in front of the commands). The startup files are identical on both hard disk and diskette, with the exception of the PATH command.

All the following copying is easier to do with a program like NC, but here are the DOS commands so that nobody feels left out. If you can boot from the hard disk, then copy the startup files over to the diskette.

```
C:\>copy config.sys a:
C:\>copy autoexec.bat a:
```

Change the `PATH` command in A:AUTOEXEC.BAT to `PATH=\;\DOS`.

(The commands given below may appear unusual to some readers because they specify a directory as the default directory *by using the CD command* but if you think about it, this is actually what CD does. The commands also make and switch to a directory in A without actually being on the A: drive. But this is perfectly acceptable.) Create the directory A:\DOS.

```
C:\>MD A:DOS
```

Set this directory as the default on A:

```
C:\>CD A:DOS
```

Set C:\DOS as the default on C:

```
C:\>CD DOS
```

Copy the files that appear in your startup files to A:\DOS. You can use F3 after every command and edit the next command a little. The first commands should look something like this:

```
C:\DOS>copy himem.sys a:
C:\DOS>copy emm386.exe a:
C:\DOS>copy display.sys a:
```

Continue until all the files mentioned in your startup files are copied.

This disk can boot your PC with the same configuration as a boot from your hard disk but without using any of the files from your hard disk. Similarly, copy these files from C:\DOS to A:\DOS: UNFORMAT.COM, FORMAT.COM, CHKDSK.EXE, (SCANDISK.EXE), UNDELETE.EXE, FDSK.EXE and SYS.COM. You might want to copy other utility programs to the diskette. Check that it works, write-protect it, keep it in a safe place – and remember where the safe place is!

It can be used in a situation where your hard disk breaks down to the extent that you cannot access or read the files on it (`cannot read drive C:` or `error reading drive C:`).

If you install a CD-ROM, sound card or a similar device, which add lines to your start files, then do not forget to copy the new start files to this boot diskette.

In addition, I would suggest you test your boot diskette at least once a month. Try it now!! It is a bit of a disaster if you really need it sometime next year and it does not work. It can happen!!

So, keep your boot diskette up to date. The work is minimal compared to the amount of time it can save you.

If disaster strikes and the only solution seems to be a repartition and/or a format or UNFORMAT of your hard disk, you can use this diskette. Fortunately, only a small minority of users ever have to face such a traumatic experience – but if you are one of them, it is no consolation knowing that you are one of only a very few. If you are fortunate enough to know someone who may be able to help you, this is a situation where you should ask for his or her assistance.

Your rescuer will be able to give you much more help if you have made one of these disks. If you are an expert who helps others, then I recommend you make such a disk yourself. Check that it works by booting from it *before* you need to use it.

Losing contact with your hard disk can also occur if something happens to your CMOS, which we describe in the next section.

CMOS and setup

(A) A safeguard, which manuals seldom describe, notes the information from what is called the CMOS (pronounced *seemos*) – the memory that holds information about a PC's configuration, including the hard disk. The information is kept "alive" by a battery that is usually rechargeable. If an accident happens and your hard disk doesn't function, and the reason may be that the CMOS has been changed or reset, you have to know the values for CMOS to get it to work again.

Fortunately, it seldom happens, and the newer your hard disk, the lower the risk. My hard disk crashed once when I was forced to press the Reset button while I was in Windows. I had no option, and crunch went my hard disk. None of my emergency Help programs could fix it. FAT, Boot Records and so on were all gone. My only course was to start again, format the hard disk, install the programs and copy my data (which I had on diskette) back to the hard disk.

With your PC, you should have received a little leaflet from the hard-disk manufacturer. If you know the type specification of your hard disk, you will find the values here. If you don't have this leaflet, ask your dealer.

You can also look into your hardware SETUP but only do this if you feel confident about what you are doing: Boot with the Reset button. Usually, it will say on screen that you can press one or more buttons to get into SETUP, where you'll find CMOS. Esc, Del or maybe Ctrl+Alt+Esc are the most common keys. If nothing appears when starting, then look up the details, which should be in your PC's manual.

You have to find CMOS setup or, if you have a menu, something about *Hard disk*. Write down the values for your hard-disk: type (usually *user-defined* no. 46 or 47) and the number of cylinders, heads and sectors. Write this information on a label and stick it on the cabinet. My hard disk has the following values: type 47, 1024 cylinders, 10 heads and 17 sectors. Usually, you need to use F10 or Esc to get out of SETUP. Occasionally you will be asked if you want to save changes. Answer `NO`.

If you are forced to say yes then do it, but be sure that you haven't changed any of the values. *If* you have or have done something and don't know the consequences, press the Reset button again. If you know that something isn't saved, then it is always safest to get out and return to the "old" state: Reset the PC!

Some setups have both standard and advanced setup. You must know what you are doing. Never change values just for fun, or to see what might happen. Your dealer really won't be very pleased with you if you do something users aren't meant to do. I know, I've tried! But generally: be careful with setup, especially advanced setup, something I am not going to deal with in this *general* text.

Touch-typing

Maybe the best advice I can give you actually has nothing directly to do with computers. I have spent years in the computer world and am constantly amazed at how few people can touch-type. I am deliberately writing this to provoke all those people who are so proud of their fast PCs, latest programs and so on. Can your turbo PC help you if it is always waiting for you to find the right key?

I spent 30 hours on an old typewriter learning to touch-type. It drove me up the wall: a s d f; l k j ... but I persevered. It really is one of those fields in which practice makes perfect. You can find books and programs that will force you to train your finger muscles.

My investment in touch-typing has been repaid many times over. I get through a lot of work, seldom hit typos. Which keyboard is the best? I don't know of a good keyboard at a fair price (IBM's are still the best but expensive).

Your health

I have placed my monitor on a low box nearly at eye-level, so I look down very slightly. If you can touch-type, you hardly ever need to look at the keyboard, which spares your neck and shoulder muscles. If you learn to touch-type, you will probably begin to make a lot of demands on your keyboard.

Turn up the contrast and brilliance on your monitor as much as is necessary for you to see sufficiently well and clearly. Windows users: Try Control Panel, Color, LCD default. Move the monitor as far away as possible – mine is 75 cm away.

Your wrists will be strained if you don't protect them. I have folded a towel several times and laid it just in front of my keyboard – a fine hand support.

The fan and the hard disk are noisy, in spite of the "Low Noise" fan regulator. I am especially sensitive to the high-frequency sound of the hard disk. Place the computer case itself as far away as possible, maybe under the desk. You can buy extension leads/cords for connecting the monitor to the computer (both for power and for data). Try blankets, towels and other things for noise insulation – without blocking the airflow, of course. There should be a little cooling of xxx, depending on your local temperature and humidity.

Find a good office chair to sit on, if you sit for long periods of time. Over one particular period, I sat for 12 hours at a time to finish a job. Don’t you try it! In the end, I became ill! Get up and move around once in a while, have a break, go for a walk, wash up, open some windows (not the program, the ones in the house with glass in them!), do some knee and arm bends, do something completely different.

Listen to your body. It doesn’t matter how infatuated you are with your computer; if your body says it needs a rest, give your body a rest. If you don’t, it will claim its revenge.

If you spend a lot of time staring at the screen, do some eye exercises once in a while. Look the whole way around to the right, then to the left, then right up and down, then down to the left, up to the left, down to the right, up to the right – alternately with eyes open and closed, and at the same time (just enough to feel your muscles) *breathe deeply*.

Now that you’ve exercised a little, it is time to look around, first with eyes closed and then with them open – kids love watching this performance. Exercises like these are very simple but have a great effect. Two minutes of exercise a day works wonders. I also have a glass screen filter, which filters out some of the harmful radiation and makes the screen image sharper and the screen anti-static. They are relatively expensive, but are probably worth the money in the long run. What is your health worth? Do not imagine that most monitor manufacturers spend much time thinking about your health, even though “green” products are starting to appear in the industry.

What do you think?

I know that you do not usually write to “an author,” but I’m simply a user who felt the urge to write this book. There are many people who know more than I do about PCs.

You are very welcome to write and tell me what you think of this guide. You are welcome to write in one of the Scandinavian languages, English or German.

With best wishes,
Michael Maardt
mm@knowware.dk