

"Computers are like Old Testament Gods; lots of rules and no mercy." - *Joseph Campbell*

Chapter 3: System Files and Commands

Boot Files	50
Optional Boot Files	53
CONFIG.SYS	54
AUTOEXEC.BAT	57
The Big Bypass	62
Loading Order	66

Technically speaking, a system file is any program file or data file that is used directly by the operating system rather than by an application. It doesn't have to end in .SYS or have a system file attribute to be considered a system file. MS-DOS and PC-DOS will often use different names for their main system files even though the functions are virtually identical. The names used here are for MS-DOS.

BOOT FILES

There are three files that are so essential to the operation of DOS, you might even say they are DOS. One key to the importance of these files is that they are involved right in the beginning as the computer is booting up and turning control of the system over to DOS. In fact, a DOS system will not successfully boot up unless these files are present on the disk, so these are sometimes called the **Boot Files**. And if you hear someone referring to **THE System Files** rather than system files in general, it's quite likely they mean these three files.

IO.SYS

The first DOS file loaded into RAM by the bootstrap loader is **IO.SYS**. As you might guess from its name, this is the file DOS uses to talk to the basic system I/O devices. Once loaded, IO.SYS loads the other two required system files plus others.

MSDOS.SYS

The MSDOS.SYS file loads next, and is really the heart of the DOS operating system. It is often referred to as the **Kernel**. This file contains all of the code for routine file management and application management.

COMMAND.COM

The **COMMAND.COM** file contains a list of the available DOS commands, and is responsible for having the right command executed in response to our typed instructions on the command line. This is also the part of DOS that puts the DOS prompt there for us. This file is often called the **DOS User Interface**, and also the **Interpreter File** because it translates our typed command into actions by the operating system.

One of the things COMMAND.COM does as soon as it's loaded is to look for a list of commands and programs that are automatically run whenever the system boots. These are found in a file called **AUTOEXEC.BAT**, which we will discuss shortly.

Many of the basic commands such as DEL and DIR are actually part of COMMAND.COM, and these are called **Internal Commands**. Commands that are not integrated into COMMAND.COM are **External Commands** and are often considered to be application programs rather than commands. EDIT and FORMAT are examples of external commands. To run an external command it must be located in the active directory or the path must include the directory where the command resides, so that COMMAND.COM knows where to find it. Internal commands are part of COMMAND.COM and will run no matter which directory is active.

Incidentally, it's only after COMMAND.COM is loaded that the system can be rebooted by pressing **Ctrl Alt Del**. And if instead of booting up you see “Bad or Missing Command Interpreter” on the screen, you will know something is amiss with COMMAND.COM.

BOOT FILE ATTRIBUTES

Remember file attributes? IO.SYS, MSDOS.SYS and COMMAND.COM are all System files, and IO.SYS and MSDOS.SYS are also Hidden files.

OPTIONAL BOOT FILES

There are two additional files which are not required for DOS to operate, but nevertheless are found on almost every DOS system. Both of these are ASCII text files that can be edited by the user, with (what else but?) the Edit command. They may also be altered by various applications, or DOS itself may ask you to add something to them.

Both files must reside in the root directory of the boot drive. Of all the system files, these two are the ones that will become most familiar to the technician.

CONFIG.SYS

The one we will discuss first, because it is loaded first, is the CONFIG.SYS file. If this file is present it will be loaded by the IO.SYS file, actually before IO.SYS loads COMMAND.COM.

DEVICE DRIVERS

CONFIG.SYS contains pointers to the **Device Drivers** and causes them to be loaded into RAM. Device drivers are system files containing instructions on how DOS can talk to the devices. These pointers in CONFIG.SYS can be recognized by the phrase `DEVICE =`. Like so:

```
DEVICE=C:\PATH\DRIVER.SYS
```

Each line in CONFIG.SYS, like the one above, is called a Statement, and each line has a number. If something is wrong with the syntax of a statement, you'll get a message something like "Error in CONFIG.SYS line XX", and you will know where to look to correct it.

By the way, if no path is stated in the statement, DOS will assume the driver is located in the root directory.

OTHER STATEMENTS

Some of the other configurations you may see in CONFIG.SYS include these:

DRIVPARM= which sets characteristics of a drive.

FILES= tells how many files DOS can be open at one time. The default is 8, but some applications may require you to increase this number.

BUFFERS= reserves up to 99 **Buffers** in RAM for disk reads. Each buffer contains 512 bytes.

DEVICEHIGH= Allows the listed driver to be loaded into the Upper Memory Area.

DOS=HIGH,UMB Allows DOS to be loaded into High Memory.

STACKS= A stack is a list of interrupts which have been called but not yet handled. Each stack has 256 bytes, and here is where you can tell DOS how much RAM to set aside for this purpose.

You may also see the **DEVICE=** statement used with terms like **HIMEM.SYS** or **EMM386.EXE**. These are used to manage memory and will be covered in the next chapter.

COMMENTING

Here's one last nifty thing you can do with **CONFIG.SYS**. If you want to put a comment in the file, for instance to remind yourself why you added or deleted a particular line, it's done by starting a line with **REM** (for **REMark**), followed by a space and then whatever you want to say.

Where **REM** comes in especially handy is when you want to take out one of the configuration statement lines momentarily just to see what happens. Put **REM** in front of the statement and DOS will see it as just a remark rather than something to do. Try your experiment, then remove the **REM** and everything is back to normal.

AUTOEXEC.BAT

This is a batch file, and contains a list of those commands and programs that the user wants the system to execute whenever it boots up. In other words, the things we want to **AUTO**matically **EXEC**ute.

AUTOEXEC.BAT is loaded and run by COMMAND.COM as one of the last actions during boot-up.

SET VARIABLES

Some of the statements you are likely to see in AUTOEXEC.BAT start with **SET=**. This is used to set a **Variable**. A variable, in case you were wondering, is any value or location which can be changed by a program or by the user. Here's an example. Most applications create temporary files (**Temp** files for short) that are only used while the application is running, and they need a place to store these. We can decide where they should put them, so this location is a variable, called **TEMP**. In use, it looks like this:

```
SET TEMP=C:\TEMP
```

With this line in AUTOEXEC.BAT, DOS will tell applications to put their temp files in a subdirectory called TEMP in the root directory of the C: drive.

ECHO

A command called **ECHO** will cause the text following the command to display on the screen, and used in AUTOEXEC.BAT, that text will display at boot-up. Try this one for grins:



```
ECHO Good Morning Handsome
```

ECHO is On by default for all commands, so these will display on the screen as they execute. If you don't want to see them, put a line in the AUTOEXEC.BAT file that says ECHO OFF.

COMMAND PATH

One very key statement used in AUTOEXEC.BAT is the **PATH** command. This tells DOS where to look for command files and executable files, any with extensions of .COM, .EXE or .BAT. On the command line we will usually name a path for the files being acted on, but not for the file containing the command itself. Well, if it's an external command and it's not in the current directory, COMMAND.COM needs to know where to find it. The PATH command can list multiple locations (separated by a semicolon) and DOS will check each location in sequence.

```
PATH=C:\DOS;A:\
```

With the above PATH= statement in AUTOEXEC.BAT, COMMAND.COM will look first in C: drive's DOS directory, and next in the root directory of the floppy A: drive.

You can use the command 'PATH' on the command line, all by itself, and it will tell you the current command path being used. For that matter many of these statements used in AUTOEXEC.BAT can also be run from the command line. You just can't do it during boot-up because the command line doesn't appear until the boot-up is complete.

REM

As in CONFIG.SYS, or any DOS program file, REM can be used to change any line of AUTOEXEC.BAT into a remark instead of an instruction.

APPLICATIONS

If you list an application in AUTOEXEC.BAT, that application will launch at boot-up. For example, many companies have employees who do nothing on their computers but order entry. With the proper line in AUTOEXEC.BAT, those systems will boot-up to the order-entry program, and the employees will never even see a DOS prompt.

Another group of programs that might be launched from AUTOEXEC.BAT are a type called **Terminate and Stay Resident**, usually abbreviated **TSR**. These are also called **Memory-resident** programs, and the idea is that they load into RAM during boot-up and stay there, to either run in the background or to be activated with a particular pattern of key-strokes. An anti-virus program is an example of a TSR.

PROMPT

In AUTOEXEC.BAT you can change the look of things, including the command-line prompt. The default prompt is created with a line saying PROMPT=\$P\$G.

BACK-UPS

It would be a good idea to make a back-up copy of AUTOEXEC.BAT, and CONFIG.SYS for that matter, any time you make changes more extensive than just remarking out a line or two. Use the COPY command and just change the extension, maybe to your initials. That way if your changes don't work you can easily go back to exactly the way it was without having to remember every tiny detail.

THE BIG BYPASS

There are occasions where you may not want the instructions in CONFIG.SYS and AUTOEXEC.BAT to be executed. An example might be with those systems we just mentioned that boot up to an order-entry program. That can be very inconvenient if you want to troubleshoot or do something other than order-entry with the system.

In DOS 6 (which is mostly what you'll see), to make the system bypass the loading of these two files all you need to do is hold down the **F5 key** while the system is booting up, after you see the message STARTING MS-DOS. You will end up at the DOS prompt just the same, but none of the instructions in CONFIG.SYS or AUTOEXEC.BAT will have been done. Of course you need to remember that the device drivers won't be loaded, maybe DOS can't find the external commands because no path was named, and you won't have specified buffers, files, stacks etc. etc.

A less drastic alternative is provided by the **F8 key**. Hold this one down during boot-up, and DOS will feed you the lines in CONFIG.SYS one at a time and ask if you want to execute each of them. Then it will give you a choice of doing the same to AUTOEXEC.BAT. If you say no, it will then bypass AUTOEXEC.BAT altogether.

If you have an earlier version of DOS you can try hitting the Ctrl and Break keys when you see AUTOEXEC.BAT start to

execute, because **Ctrl-Break** will interrupt the execution of any DOS batch file. If things get so botched up that the system won't come up, put in a bootable floppy disk so you can boot up and REM out lines in CONFIG.SYS and AUTOEXEC.BAT until you find the offenders. Tedious yes, but also tried and true. I recommend adding a DOS-bootable floppy to your toolkit.

ASSORTED SYSTEM COMMANDS

So far we have looked at commands for file and directory management, and some of the more common ones used in the start-up files. There are some more you should know that are just general system commands.

VER - This one tells you the version of DOS being used. At least, it provides the version that DOS has listed in a file called the **Application Table**. Before Version 5, the DOS version on the system was fixed and discrepancies were highly unlikely.

SETVER - DOS Version 5 provided this command, which allowed the user to update the DOS version listed in the application table. With this same DOS 5, users were able for the first time to upgrade their DOS. Coincidence, or not?

DATE - Changes the date value in the DOS date/time register. If entered without a date value, it will return the current date setting.

TIME - This does for the time value what DATE does for the date value.

PRINT - Sends a file to the printer. Are you surprised?

MODE - This command can set parameters for the printer, keyboard and other peripherals.

LOADHIGH - Loads TSRs into upper memory. The short version of this command is LH.

MSAV - This is a virus-check program available in DOS 6. It runs each time it's called up.

VSAFE - Another anti-virus program in DOS 6. This one runs continuously in the background.

MSD - This is a diagnostic utility built into DOS versions 5 and later. It will tell you about your ports and various peripherals, how many drives, how much memory and other nifty facts like these.

PROMPT - Here you can customize the DOS prompt.

QBASIC - Puts you into the QBASIC program-writer utility, whenever you get an overwhelming urge to write your own DOS programs.

DOSKEY - Remember the trick of enabling the up-arrow to go back to previous command lines? This is usually run from AUTOEXEC.BAT so you don't have to enter it yourself after every boot-up.

DOSSHELL - A shell is a program that is layered over another program. This one is put on top of the basic DOS program so that we can use a mouse to make screen selections, pull down menus and move the cursor around to our heart's content. Most people don't want to go back once they've used the shell, so they put this command into AUTOEXEC.BAT. To run the shell in text mode, use /T, and /G for graphics mode.

LOADING ORDER

From the earlier discussion of the boot files you may have been able to figure out which order they loaded in, but we never said it in so many words and it is something you should know. Here it is:

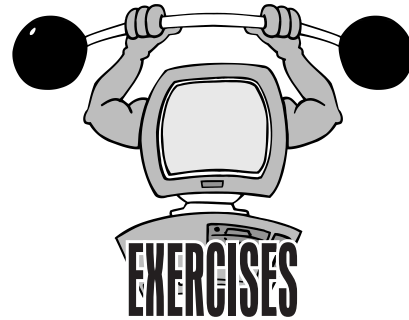
IO.SYS
MSDOS.SYS
CONFIG.SYS
COMMAND.COM
AUTOEXEC.BAT

So there you have it. And one more little point I'll sneak in here. Occasionally you may not be sure whether to execute a file from CONFIG.SYS or from AUTOEXEC.BAT, so here's a rule of thumb.

If the file has an extension of .COM it should go in AUTOEXEC.BAT. Remember, the statements in AUTOEXEC.BAT are executed as commands, by COMMAND.COM.

If the file has an extension of .SYS, put it in CONFIG.SYS along with the device drivers, which are all .SYS files. If it has some other extension such as .EXE, ask yourself if it interprets for the OS (like a CONFIG.SYS device driver) or is it something for the OS to execute (AUTOEXEC.BAT)?

EXERCISE #2: MORE DOS



1. Go to the hard drive's root directory and enter the command TYPE CONFIG.SYS. Use /P if needed.

2. How many device drivers are listed? _____ .

Are they loaded into upper or conventional memory?

3. Now look at AUTOEXEC.BAT.

Is there a PATH= statement? Where does it point to?

4. Are there any statements that load TSRs? _____

Do they load into upper or conventional memory?

CHAPTER QUIZ

MULTIPLE CHOICE

Circle the best answer for each statement.

1. What key allows bypassing CONFIG.SYS and AUTOEXEC.BAT during boot-up?
 - a. F1
 - b. F5
 - c. F8
 - d. Escape

2. What key allows step-by-step confirmation of CONFIG.SYS and AUTOEXEC.BAT?
 - a. F5
 - b. F8
 - c. F11
 - d. Ctrl

3. What does TSR stand for?
 - a. Terminal Set Ready
 - b. Terminate and Stay Resident
 - c. Testing System Ready
 - d. Temptation Still Ripe

4. What executes the statements in AUTOEXEC.BAT?
 - a. IO.SYS
 - b. MSDOS.SYS
 - c. COMMAND.COM
 - d. CONFIG.SYS

5. PATH = will be found in which file?
 - a. COMMAND.COM
 - b. CONFIG.SYS
 - c. AUTOEXEC.BAT
 - d. MSDOS.SYS

6. You've just upgraded DOS and an old application won't run. What do you do?
 - a. Run the SETVER command.
 - b. Delete the application.
 - c. Reinstall the application.
 - d. Reinstall the DOS upgrade.

CHAPTER 3
