THE PROGRAMMER'S TECHNICAL REFERENCE:

MS-DOS, IBM PC & Compatibles

THE PROGRAMMER'S TECHNICAL REFERENCE:

MS-DOS, IBM PC & Compatibles

Dave Williams

SIGMA PRESS – Wilmslow, United Kingdom

First published in 1990 by

Sigma Press, 1 South Oak Lane, Wilmslow, Cheshire SK9 6AR, England.

British Library Cataloguing in Publication Data

A CIP catalogue record for this book is available from the British Library.

ISBN: 1-85058-199-1

Typesetting and design by

Sigma Hi-Tech Services Ltd

Printed in Malta by

Interprint Ltd.

Distributed by

John Wiley & Sons Ltd., Baffins Lane, Chichester, West Sussex, England.

Acknowledgement of copyright names

Within this book, various proprietary trade names and names, as listed below, are protected by copyright and are mentioned for descriptive purposes:

UNIX, *AT&T*, Allied Telephone and Telegraph; *AST*, *RAMpage!* AST Corporation; *Atari*, *ST*, Atari Computer; *Borland*, *Turbo C*, *Turbo Pascal*, *Turbo Lightning*, Borland; *Amiga 2000*, Commodore Business Machines; *Compaq*, *Deskpro*, Compaq Computer Corporation; *Corona*, *Cordata*, *Cordata Computer*; *10-Net*, Fox Research, Inc.; *Smartmodem*, Hayes; *IBM*, *PC*, *PCjr*, *PC/XT*, *PC/AT*, *XT/286*, *PS/2*, *TopView*, *DOS*, *PC-DOS*, *Micro Channel 3270 PC*, *RT PC*, *Token Ring*, IBM Corporation; *Intel*, *iAPX286*, *iAPX386*, *LIM EMS*, *Communicating Applications Standard*, Intel Corporation; *Logitech*, *Logimouse*, Logitech, Inc.; *Microsoft*, *MS*, *MS-DOS*, *OS/2*, *Xenix*, *Windows*, *Windows/286*, *Windows/386*, *Microsoft Networks*, *LIM EMS*, *XMA*, Microsoft Corp.; *Mouse Systems*, Mouse Systems Corp.; *Novell*, *NetWare*, Novell Corp.; *DesQview*, Quarterdeck Office Systems; *ARC*, SEAware, Inc.; *DoubleDOS*, Softlogic; *TaskView*, Sunny Hill Software; *Tandy*, Tandy Corp.; *Zenith*, *Z-100*, Zenith Radio Corporation; *ShowPartner*, *Paintbrush*, ZSoft Corporation; *'LIM 4.0'* and *'Expanded Memory Specification'* are copyright Lotus Development Corp, Intel Corp, and Microsoft Corp; *'EEMS'*, *'AQA 3.1'* and *'Enhanced Expanded Memory Specification'* are copyright by Ashton-Tate, Quadram, and AST.Various other names are trademarks of their respective companiesFull acknowledgment is hereby made of all such protection.

Preface

This book is a technical reference. It is NOT a tutorial. It is intended to replace the various (expensive) references needed to program for the DOS environment, that stack of magazines threatening to take over your work area, and those odd tables and charts you can never find when you need them.

The various Microsoft and IBM publications and references don't always have the same information. This has caused some consternation about the 'undocumented' features to be found in DOS. In general, if a call doesn't appear in the IBM DOS Technical Reference it is considered 'undocumented' although it may be in common use.

Microsoft's offical policy toward DOS has been to put the burden of documenting and supporting their product to their vendors. Microsoft will not answer any questions concerning DOS directly since they don't officially support it. This leaves what information IBM and other OEMs (DEC, Zenith, et al) have chosen to publish, and the information obtained from programmers who've poked around inside it.

Now that Microsoft is selling MSDOS 3.3 and 4.0 over the counter they seem to be dragging their feet over whether they will have to support the generic version since it doesn't have an OEM name on it anymore. In view of their push to OS/2 (OS/2! Just Say No!) further support of DOS seems unlikely.

A project this size takes a LOT of time and effort. I've tried to verify as much of the information I've received as I could, but there's just too much for absolute certainty.

Contents

1

DOS and the IBM PC

Some History

Development of MS-DOS/PCDOS began in October 1980, when IBM began searching the market for an operating system for the yet-to-be-introduced IBM PC. Microsoft had no real operating system to sell, but after some research licensed Seattle Computer Products' 86-DOS operating system, which had been written by a man named Tim Paterson earlier in 1980 for use on that company's line of 8086, S100 bus micros. 86-DOS (also called QDOS, for Quick and Dirty Operating System) had been written as more or less a 16-bit version of CP/M, since Digital Research was showing no hurry in introducing CP/M-86.

This code was hurriedly polished up and presented to IBM for evaluation. IBM had originally intended to use Digital Research's CP/M operating system, which was the industry standard at the time. Folklore reports everything from obscure legal entanglements to outright snubbing of the IBM representatives by Digital. Irregardless, IBM found itself left with Microsoft's offering of "Microsoft Disk Operating System 1.0". An agreement was reached between the two, and IBM agreed to accept 86-DOS as the main operating system for their new PC. Microsoft purchased all rights to 86-DOS in July 1981, and "IBM PC-DOS 1.0" was ready for the introduction of the IBM PC in October 1981. IBM subjected the operating system to an extensive quality-assurance program, reportedly found well over 300 bugs, and decided to rewrite the programs. This is why PC-DOS is copyrighted by both IBM and Microsoft.

It is sometimes amusing to reflect on the fact that the IBM PC was not originally intended to run MS-DOS. The target operating system at the end of the development was for a (not yet in existence) 8086 version of CP/M. On the other hand, when DOS was originally written the IBM PC did not yet exist! Although PC-DOS was bundled with the computer, Digital Research's CP/M-86 would probably have been the main operating system for the PC except for two things - Digital Research wanted $495 for CP/M-86 (considering PC-DOS was essentially free) and many software developers found it easier to port existing CP/M software to DOS than to the new version of CP/M. Several computer magazines claimed that Digital Research aided IBM in writing DOS 4.0, which was subsequently licensed back to Microsoft, which has dropped further development of the operating system to tilt at the windmills of OS/2. OS/2? Not yet! After using DR-DOS 3.4 and noting its behaviour, I now tend to seriously doubt Digital had any dealings with PC-DOS 4.0.

MS-DOS and PC-DOS have been run on more than just the IBM-PC and clones. Some of the following have been done:

Hardware PC Emulation:	
Commodore Amiga 2000	8088 or A2286D 80286 Bridge Board
IBM PC/AT	80286 AT adapter
Atari 400/800	Co-Power 88 board
Apple Macintosh	AST 80286 board
Atari ST	PC-Ditto II cartridge
Apple II	TransPC 8088 board, QuadRam QuadLink
Software PC Emulation:	
Atari ST	PC-Ditto I
Apple Macintosh	SoftPC
DOS Emulation:	
OS/2	DOS emulation in "Compatibility Box"
QNX	DOS window
SunOS	DOS window
Xenix	DOS emulation with DOSMerge

What is DOS?

DOS exists as a high-level interface between an application program and the computer. DOS stands for "Disk Operating System", which reflects the fact that its main original purpose was to provide an interface between the computer and its disk drives.

DOS now lets your programs do simple memory management, I/O from the system console, and assorted system tasks (time and date, etc) as well as managing disk operations. Versions 3.1 and up also incorporate basic networking functions.

With the introduction of installable device drivers and TSR (terminate but stay resident) programs in DOS 2.0, the basic DOS functions may be expanded to cover virtually any scale of operations required.

Other Operating Systems

There are a number of compatible replacements for Microsoft's MS-DOS. Some are:

Consortium Technologies MultiDOS	(multitasking, multiuser)
Digital Research Concurrent DOS	(multitasking)
Digital Research Concurrent DOS 386	(for 80386 computers)
Digital Research Concurrent DOS XM	(multitasking, multiuser)
Digital Research DR-DOS 3.31 and 4.0	(PC-DOS clones)
PC-MOS/386	(multitasking, multiuser)
Wendin-DOS	(multitasking, multiuser)
VM/386	(multitasking)

Various other operating systems are available for the IBM PC. These include:

Digital Research CP/M-86
Digital Research Concurrent CP/M-86 (multitasking)
Minix (multitasking UNIX workalike)
Pick (database-operating system)

QNX (multitasking, multiuser)
UNIX (various systems from IBM itself, Microsoft-SCO, Bell, and various UNIX clones, single and multi user) (AIX, Xenix, AT&T System V, etc.)

"Shell" programs exist which use DOS only for disk management while they more or less comprise a new operating system. These include:

DesQview
Windows
OmniView
GEM
TopView
TaskView

Specific Versions of MS/PC-DOS

DOS 1.x is essentially 86-DOS. DOS 2.x kept the multiple file layout (the two hidden files and COMMAND.COM) but for all practical purposes is an entirely different operating system with backwards compatibility with 1.x. I seriously doubt there has been much code from 1.x retained in 2.x. DOS 3.x is merely an enhancement of 2.x; there seems little justification for jumping a whole version number. DOS 4.0, originating as it did from outside Microsoft, can justify a version jump. Unfortunately, 4.x seems to have very little reason to justify its existence - virtually all of its core features can be found in one version or another of DOS 3.x.

DOS version nomenclature: major.minor.minor. The digit to the left of the decimal point indicates a major DOS version change. 1.0 was the first version. 2.0 added support for subdirectories, 3.0 added support for networking, 4.0 added some minimal support for Lotus-Intel-Microsoft EMS.

The first minor version indicates customization for a major application. For example, 2.1 for the PCjr, 3.3 for the PS/2s. The second minor version does not seem to have any particular meaning.

The main versions of DOS are:

PC-DOS 1.0	August 1981	original release
PC-DOS 1.1	May 1982	bugfix, double sided drive support
MS-DOS 1.25	June 1982	for early compatibles
PC-DOS 2.0	March 1983	for PC/XT, Unix-type subdirectory support
PC-DOS 2.1	October 1983	for PCjr, bugfixes for 2.0
MS-DOS 2.11	October 1983	compatible equivalent to PC-DOS 2.1
PC-DOS 3.0	August 1984	1.2 meg drive for PC/AT, some new system calls
PC-DOS 3.1	November 1984	bugfix for 3.0, implemented network support
MS-DOS 2.25	October 1985	compatible; extended foreign language support
PC-DOS 3.2	December 1985	720k 3.5 inch drive support for Convertible
PC-DOS 3.3	April 1987	for PS/2 series, 1.44 meg, multiple DOS partitions
MS-DOS 3.31	November 1987	over-32 meg DOS partitions, new function calls
PC-DOS 4.0	August 1988	minor EMS support, some new function calls
MS-DOS 4.01	January 1989	Microsoft version with some bugfixes

IBM's PC-DOS is considered to be the "standard" version of DOS; Microsoft has sold MS-DOS over the counter only since version 3.2 (previously, Microsoft sold its versions only to OEMs).

Most versions of DOS functionally duplicate the external DOS commands such as DISKCOPY, etc. Although Microsoft announced that they would sell MS-DOS 4.0 only to OEMs, they apparently changed the policy and are now selling it over the counter.

Some versions of MS-DOS varied from PC-DOS in the available external commands. Some OEMs only licensed the basic operating system code (the xDOS and xBIO programs, and COMMAND.COM) from Microsoft, and either wrote the rest themselves or contracted them from outside software houses like Phoenix. Most of the external programs for DOS 3.x and 4.x are written in "C" while the 1.x and 2.x utilities were written in assembly language. Other OEMs required customized versions of DOS for their specific hardware configurations, such as Sanyo 55x and early Tandy computers, which were unable to exchange their DOS with the IBM version.

At least two versions of DOS have been modified to be run entirely out of ROM. The Sharp PC5000 had MS-DOS 1.25 in ROM, and the Toshiba 1000 and some Tandy 1000 models have MS-DOS 2.11 in ROM. Digital Research has also announced its DR-DOS is available in a ROM version and Award Software is marketing DOS cards to OEMs as a plug-in.

PC-DOS 3.0 was extremely buggy on release. It does not handle the DOS environment correctly and there are numerous documented problems with the batch file parser. The network support code is also nonfunctional in this DOS version. It is recommended that users upgrade to at least version 3.1.

DEC MS-DOS versions 2.11 for the Rainbow had the ANSI.SYS device driver built into the main code. The Rainbow also used a unique quad density, single-sided floppy drive and its DOS had special support for it.

IBM had a version 1.85 of PC-DOS in April 1983, after the introduction of DOS 2.0. It was evidently for internal use only, supported multiple drive file searches (a primitive form of PATH), built in MODE commands for screen support, a /P parameter for TYPE for paused screens, an editable command stack like the public domain DOSEDIT.COM utility, and could be set up to remain completely resident in RAM instead of a resident/transient part like normal DOS. It is a pity some of the neat enhancements didn't make it into DOS 2.0. IBM also had an "internal use only" version 3.4, evidently used while developing DOS 4.0.

Some versions of DOS used in compatibles do not maintain the 1.x, 2.x, ... numbering system. Columbia Data Products computers labelled DOS 1.25 as DOS 2.0. Early Compaqs labelled DOS 2.0 as DOS 1.x. Other versions incorporated special features - Compaq DOS 3.31 and Wyse DOS 3.21 both support 32-bit file allocation tables in the same fashion as DOS 4.x.

According to PC Week Magazine, July 4, 1988. Arabic versions of MS-DOS are shipping with a hardware copy-protection system from Rainbow Technologies. This is similar to the short-lived system used by AutoCAD 2.52 and a very few other MS-DOS programs, where an adapter block is plugged into the parallel port and software makes use of coded bytes within the block. This type of copy protection has been common on Commodore products for several years, where it is called a "dongle".

The AutoCAD dongle was defeated by a small program written within weeks of version 2.52's debut. Version 2.62 was released 3 months later, without the dongle. The DOS dongle will, however, prevent the system from booting at all unless it is found.

This makes the Arabic version of MS-DOS the first copy-protected operating system, a dubious distinction at best. The modifications to the operating system to support the dongle are not known at this time. Frankly, it would seem that burning the operating system into ROMs would be cheaper and simpler.

Versions of DOS sold in Great Britain are either newer than those sold in the US or use a different numbering system. DOS 3.4, 4.0, 4.1, 4.2, and 4.3 had been released here between the US releases of 3.3 and 4.0.

Microsoft changed their OEM licensing agreements between DOS versions 2.x and 3.x. OEM versions of DOS 3.x must maintain certain data areas and undocumented functions in order to provide compatibility with the networking features of the operating system. For this reason, resident programs will be much more reliable when operating under DOS 3.x.

IBM's release of DOS 4.0 (and the immediate subsequent release of a bugfix) is a dubious step "forward". DOS 4.0 is the first version of DOS to come with a warranty; the catch is that IBM warrants it only for a very slim list of IBM-packaged software. 4.0 has some minor EMS support, support for large hard disks, and not much else. With its voracious RAM requirements and lack of compatibility with previous versions of DOS (many major software packages crash under DOS 4.0), plus the increase in price to a cool $150, there has been no great rush to go to the newest DOS

The Operating System Hierarchy

The Disk Operating System (DOS) and the ROM BIOS serve as an insulating layer between the application program and the machine, and as a source of services to the application program.

As the term 'system' might imply, DOS is not one program but a collection of programs designed to work together to allow the user access to programs and data. Thus, DOS consists of several layers of "control" programs and a set of "utility" programs.

The system hierarchy may be thought of as a tree, with the lowest level being the actual hardware. The 8088 or V20 processor sees the computer's address space as a ladder two bytes wide and one million bytes long. Parts of this ladder are in ROM, parts in RAM, and parts are not assigned. There are also various "ports" that the processor can use to control devices.

The hardware is normally addressed by the ROM BIOS, which will always know where everything is in its particular system. The chips may usually also be written to directly, by telling the processor to write to a specific address or port. This sometimes does not work as the chips may not always be at the same addresses or have the same functions from machine to machine.

DOS Structure

DOS consists of four components.

The boot record
The ROM BIOS interface (IBMBIO.COM or IO.SYS)
The DOS program file (IBMDOS.COM or MS-DOS.SYS)
The command processor (COMMAND.COM or aftermarket replacement)

The Boot Record

The boot record begins on track 0, sector 1, side 0 of every diskette formatted by the DOS FORMAT command. The boot record is placed on diskettes to produce an error message if you try to start up the system with a non-system diskette in drive A. For hard disks, the boot record resides

on the first sector of the DOS partition. All media supported by DOS use one sector for the boot record.

Read Only Memory (ROM) BIOS Interface and Extensions

The file IBMBIO.COM or IO.SYS is the interface module to the ROM BIOS. This file provides a low-level interface to the ROM BIOS device routines and may contain extensions or changes to the system board ROMs. Some compatibles do not have a ROM BIOS to extend, and load the entire BIOS from disk (Sanyo 55x, Viasyn machines). Some versions of MS-DOS, such as those supplied to Tandy, are named IBMBIO.COM but are not IBM files.

These low-level interface routines include the instructions for performing operations such as displaying information on the screen, reading the keyboard, sending data out to the printer, operating the disk drives, and so on. It is the operating system's means of controlling the hardware. IBMBIO.COM contains any modifications or updates to the ROM BIOS that are needed to correct any bugs or add support for other types of hardware such as new disk drives. By using IBMBIO.COM to update the ROM BIOS on the fly when the user turns on their computer, IBM does not need to replace the ROM BIOS chip itself, but makes any corrections through the cheaper and easier method of modifying the IBMBIO.COM file instead.

IBMBIO.COM also keeps track of hardware operations on an internal stack or "scratch pad" area for the operating system to save information such as addresses it will need, etc. An example of the use for this stack can be seen when running a program such as a word processor. If you have told the word processor to save your letter, it will write the data to your disk. During this time, if you start typing some more information, the keyboard generates a hardware interrupt. Since you don't want the process of writing the information to the disk to be interrupted, DOS allocates a slot in the stack for the keyboard's hardware interrupt and when it gets a chance, (probably after the data has been written to the disk), it can process that interrupt and pick up the characters you may have been typing. The STACKS= command in DOS 3.2+'s CONFIG.SYS file controls the number of stack frames available for this purpose.

IBMBIO.COM also reads your CONFIG.SYS file and installs any device drivers (i.e. DEVICE=ANSI.SYS) or configuration commands it may find there.

The DOS Program

The actual DOS program is the file IBMDOS.COM or MS-DOS.SYS. It provides a high-level interface for user (application) programs. This program consists of file management routines, data blocking/deblocking for the disk routines, and a variety of built-in functions easily accessible by user programs.

When a user program calls these function routines, they accept high-level information by way of register and control block contents. When a user program calls DOS to perform an operation, these functions translate the requirement into one or more calls to IBMBIO.COM, MS-DOS.SYS or system hardware to complete the request.

The Command Interpreter

The command interpreter, COMMAND.COM, is the part you interact with on the command line. COMMAND.COM has three parts. IBM calls them the "resident portion", the "initialization portion" and the "transient portion".

IBM's original documentation spoke of installing alternate command interpreters (programs other than COMMAND.COM) with the SHELL= statement in CONFIG.SYS. Unfortunately, IBM chose not to document much of the interaction between IBMDOS.COM and IBMBIO.COM. By the time much of the interaction was widely understood, many commercial software programs had been written to use peculiarities of COMMAND.COM itself.

Two programs exist that perform as actual "shells" by completely replacing COMMAND.COM and substituting their own command interpreter to use with the hidden DOS files. These are Command Plus, a commercial package, and the very interesting shareware 4DOS package. Both supply greatly enhanced batch language and editing capabilities.

Note: DOS 3.3+ checks for the presence of a hard disk, and will default to COMSPEC=C:\. Previous versions default to COMSPEC=A:\. Under some DOS versions, if COMMAND.COM is not immediately available for reloading (i.e., swapping to a floppy with COMMAND.COM on it) DOS may crash.

Resident Portion

The resident portion resides in memory immediately following IBMDOS.COM and its data area. This portion contains routines to process interrupts 22h (Terminate Address), 23h (Ctrl-Break Handler), and 24h (Critical Error Handler), as well as a routine to reload the transient portion if needed. For DOS 3.x, this portion also contains a routine to load and execute external commands, such as files with extensions of COM or EXE.

When a program terminates, a checksum is used to determine if the application program overlaid the transient portion of COMMAND.COM. If so, the resident portion will reload the transient portion from the area designated by COMSPEC= in the DOS environment. If COMMAND.COM cannot be found, the system will halt.

All standard DOS error handling is done within the resident portion of COMMAND.COM. This includes displaying error messages and interpreting the replies to the "Abort, Retry, Ignore, Fail?" message.

Since the transient portion of COMMAND.COM is so large (containing the internal commands and all those error messages), and it is not needed when the user is running an application it can be overlaid that program if that application needs the room. When the application is through, the resident portion of COMMAND.COM brings the transient portion back into memory to show the prompt. This is why you will sometimes see the message "Insert disk with COMMAND.COM". It needs to get the transient portion off the disk since it was overlaid with the application program.

The initialization portion of COMMAND.COM follows the resident portion and is given control during the boot-up procedure. This section actually processes the AUTOEXEC.BAT file. It also decides where to load the user's programs when they are executed. Since this code is only needed during start-up, it is overlaid by the first program which COMMAND.COM loads.
The transient portion is loaded at the high end of memory and it is the command processor itself. It interprets whatever the user types in at the keyboard, hence messages such as 'Bad command or file name' for when the user misspells a command. This portion contains all the internal commands (i.e. COPY, DIR, RENAME, ERASE), the batch file processor (to run .BAT files) and a routine to load and execute external commands which are either .COM or .EXE files.

The transient portion of COMMAND.COM produces the system prompt, (C), and reads what

the user types in from the keyboard and tries to do something with it. For any .COM or .EXE files, it builds a command line and issues an EXEC function call to load the program and transfer control to it.

DOS Initialization

The system is initialized by a software reset (Ctrl-Alt-Del), a hardware reset (reset button), or by turning the computer on. The Intel 80x8x series processors always look for their first instruction at the end of their address space (0FFFF0h) when powered up or reset. This address contains a jump to the first instruction for the ROM BIOS.

Built-in ROM programs (Power-On Self-Test, or POST, in the IBM) check machine status and run inspection programs of various sorts. Some machines set up a reserved RAM area with bytes indicating installed equipment (AT and PCjr).

When the ROM BIOS finds a ROM on an adapter card, it lets that ROM take control of the system so that it may perform any set up necessary to use the hardware or software controlled by that ROM. The ROM BIOS searches absolute addresses 0C8000h through 0E0000h in 2K increments in search of a valid ROM. A valid ROM is determined by the first few bytes in the ROM. The ROM will have the bytes 55h, 0AAh, a length indicator and then the assembly language instruction to CALL FAR (to bring in a 'FAR' routine). A checksum is done on the ROM to verify its integrity, then the BIOS performs the CALL FAR to bring in the executable code. The adapter's ROM then performs its initialization tasks and hopefully returns control of the computer back to the ROM BIOS so it can continue with the booting process.

The ROM BIOS routines then look for a disk drive at A: or an option ROM (usually a hard disk) at absolute address C:800h. If no floppy drive or option ROM is found, the BIOS calls int 19h (ROM BASIC if it is an IBM) or displays an error message.

If a bootable disk is found, the ROM BIOS loads the first sector of data from the disk and then jumps into the RAM location holding that code. This code normally is a routine to load the rest of the code off the disk, or to 'boot' the system.

The following actions occur after a system initialization:

1. The boot record is read into memory and given control.
2. The boot record then checks the root directory to assure that the first two files are IBMBIO.COM and IBMDOS.COM. These two files must be the first two files, and they must be in that order (IBMBIO.COM first, with its sectors in contiguous order).
   Note: IBMDOS.COM need not be contiguous in version 3.x+.
3. The boot record loads IBMBIO.COM into memory.
4. The initialization code in IBMBIO.COM loads IBMDOS.COM, determines equipment status, resets the disk system, initializes the attached devices, sets the system parameters and loads any installable device drivers according to the CONFIG.SYS file in the root directory (if present), sets the low-numbered interrupt vectors, relocates IBMDOS.COM downward, and calls the firstbyte of DOS.
   Note: CONFIG.SYS may be a hidden file.
5. DOS initializes its internal working tables, initializes the interrupt vectors for interrupts 20h through 27h, and builds a Program Segment Prefix for COMMAND.COM at the lowest available segment. For DOS versions 3.10 up, DOS also initializes the vectors for interrupts

0Fh through 3Fh. An initialization routine is included in the resident portion and assumes control during start-up. This routine contains the AUTOEXEC.BAT file handler and determines the segment address where user application programs may be loaded. The initialization routine is then no longer needed and is overlaid by the first program COMMAND.COM loads.
Note: AUTOEXEC.BAT may be a hidden file.

6. IBMBIO.COM uses the EXEC function call to load and start the top-level command processor. The default command processor is COMMAND.COM in the root directory of the boot drive. If COMMAND.COM is in a subdirectory or another command processor is to be used, it must be specified by a SHELL= statement in the CONFIG.SYS file. A transient portion is loaded at the high end of memory. This is the command processor itself, containing all of the internal command processors and the batch file processor. For DOS 2.x, this portion also contains a routine to load and execute external commands, such as files with extensions of COM or EXE. This portion of COMMAND.COM also produces the DOS prompt (such as 'A'), reads the command from the standard input device (usually the keyboard or a batch file), and executes the command. For external commands, it builds a command line and issues an EXEC function call to load and transfer control to the program.

Note 1. COMMAND.COM may be a hidden file.

2. For IBM DOS 2.x, the transient portion of the command processor contains the EXEC routine that loads and executes external commands. For MS-DOS 2.x+ and IBM DOS 3.x+, the resident portion of the command processor contains the EXEC routine.

3. IBMBIO only checks for a file named COMMAND.COM. It will load any file of that name if no SHELL= command is used.

That pretty much covers the boot-up process. After COMMAND.COM is loaded, it runs the AUTOEXEC.BAT file and then the user gets a prompt to begin working.

2

CPU Port Assignments, System Memory Map, BIOS Data Area, Interrupts 00h to 09h

Introduction

For consistency in this reference, all locations and offsets are in hexadecimal unless otherwise specified. All hex numbers are prefaced with a leading zero if they begin with an alphabetic character, and are terminated with a lowercase H (h). The formats vary according to common usage.

System Memory Map

The IBM PC handles its address space in 64k segments, divided into 16k fractions and then further as necessary.

start addr. (dec)	start addr. (hex)	end addr	usage
	640k RAM Area		
0k			start of RAM, first K is interrupt vector table
16k	00000-03FFF		PC-0 system board RAM ends
32k	04000-07FFF		
48k	08000-0BFFF		
64k	10000-13FFF		PC-1 system board RAM ends
80k	14000-17FFF		
96k	18000-1BFFF		
112k	1C000-1FFFF		
128k	20000-23FFF		
144k	24000-27FFF		
160k	28000-2BFFF		
176k	2C000-2FFFF		
192k	30000-33FFF		
208k	34000-37FFF		
224k	38000-3BFFF		
240k	3C000-3FFFF		

```
256k  40000-43FFF   PC-2 system board RAM ends
272k  44000-47FFF
288k  48000-4BFFF
304k  4C000-4FFFF

320k  50000-53FFF
336k  54000-57FFF
352k  58000-5BFFF
368k  5C000-5FFFF

384k  60000-63FFF
400k  64000-67FFF
416k  68000-6BFFF
432k  6C000-6FFFF

448k  70000-73FFF
464k  74000-77FFF
480k  78000-7BFFF
496k  7C000-7FFFF

512k  80000-83FFF
528k  84000-87FFF
544k  88000-8BFFF   the original IBM PC-1 BIOS limited memory to 544k
560k  8C000-8FFFF

576k  90000-93FFF
592k  94000-97FFF
609k  98000-9BFFF
624k  9C000-9FFFF   to 640k (top of RAM address space)

 A0000 ***** 64k ***** EGA address

640k  A0000-A95B0   MCGA 320x200 256 colour video buffer
          -AF8C0    MCGA 640x480 2 colour video buffer
          -A3FFF
656k  A4000-A7FFF
672k  A8000-ABFFF       this 64k segment may be used for contiguous DOS
688k  AC000-AFFFF       RAM with appropriate hardware and software

 B0000 ***** 64k ***** mono and CGA address

704k  B0000-B3FFF  4k  monochrome display   | PCjr and early Tandy 1000
720k  B4000-B7FFF                           | BIOS revector direct write to the
                                            | B8 area to the Video Gate Array
736k  B8000-BBFFF  16k CGA uses             | and reserved system RAM
756k  BC000-BFFFF

 C0000 ***** 64k ********** expansion ROM

768k  C0000-C3FFF  16k EGA BIOS C000:001E EGA BIOS signature (letters IBM
784k  C4000-C5FFF
      C6000-C63FF  256 bytes Professional Graphics Display comm. area
      C6400-C7FFF
800k  C8000-CBFFF  16k hard disk controller BIOS, drive 0 default
      CA000            some 2nd floppy (high density) controller BIOS
816k  CC000-CDFFF   8k IBM PC Network NETBIOS
      CE000-CFFFF

 D0000 ***** 64k ***** expansion ROM

832k  D0000-D7FFF  32k IBM Cluster Adapter  | PCjr first ROM cartridge
          DA000    voice communications     | address area.
848k  D4000-D7FFF                           | Common expanded memory
864k  D8000-DBFFF                           | board paging area.
880k  DC000-DFFFF

 E0000 ***** 64k ***** expansion ROM

896k  E0000-E3FFF                           | PCjr second ROM cartidge
912k  E4000-E7FFF                           | address area
928k  E8000-EBFFF
```

```
944k  EC000-EFFFF                              |    spare ROM sockets on AT

 F0000 ***** 64k ***** system

960k  F0000-F3FFF  reserved by IBM             |    cartridge address
976k  F4000-                                   |    area (PCjr cartridge
      F6000        ROM BASIC Begins            |    BASIC)
992k  F8000-FB000                              |
1008k FC000-FFFFF  ROM BASIC and original      |
                   BIOS (Compatibility BIOS    |
                   in PS/2)                    |
1024k       FFFFF  end of memory (1024k) for 8088 machines

384k  100000-15FFFF 80286/AT extended memory area, 1Mb motherboard
15Mb  100000-FFFFFF 80286/AT extended memory address space
15Mb  160000-FDFFFF Micro Channel RAM expansion (15Mb extended memory)
128k  FE0000-FFFFFF system board ROM            (PS/2 Advanced BIOS)
```

Note that the ROM BIOS has a duplicated address space which causes it to 'appear' both at the end of the 1 megabyte real mode space and at the end of the 16 megabyte protected mode space. The addresses from 0E0000 to 0FFFFF are equal to 0FE0000 to 0FFFFFF. This is necessary due to differences in the memory addressing between Real and Protected Modes.

PC Port Assignment

hex address	Function	Models

```
                                                PCjr |PC  |XT  |AT  |CVT |M30 |PS2
0000-000F   8237 DMA controller                      PC
0010-001F   8237 DMA controller                                 AT              PS2
0020-0027   8259A interrupt controller
0020-003F   8259A interrupt controller (AT)
0020-0021   Interrupt controller 1, 8259A            PC         AT              PS2
0040-0043   Programmable timer 8253                  PC
0040-0047   Programmable timers                                                  PS2
0040-005F   8253-5 programmable timers                          AT
            (note: 0041 was memory refresh in PCs. Not used in PS/2)
0060-0063   Keyboard controller 8255A                PC
0060-006F   8042 keyboard controller                            AT
0060        IOSGA keyboard input port                                           PS2
0061        speaker                          PCjr  PC  XT  AT  CVT
0061        IOSGA speaker control                                        M30    PS2
0061        On some clones, setting or clearing bit 2 controls Turbo mode
0062        IOSGA configuration control                                  M30    PS2
0063        SSGA, undocumented                                                  PS2
0064        keyboard auxiliary device                                           PS2
0065-006A   SSGA, undocumented                                                 PS2
006B        SSGA, RAM enable/remap                                              PS2
006C-006F   SSGA, undocumented                                                 PS2
0070        AT CMOS write internal register
0071        AT CMOS read internal register
0070-0071   CMOS real-time clock, NMI mask                                      PS2
0070-007F   CMOS real-time clock, NMI mask                      AT
0074-0076   reserved                                                           PS2
0800-008F   SSGA DMA page registers                                            PS2
0080-009F   DMA page registers, 74LS612                         AT
0090        central arbitration control port                    (Micro Channel)
0091        card selected feedback                              (Micro Channel)
0092        system control port A                               (Micro Channel)
0093        reserved                                            (Micro Channel)
0094        system board setup                                  (Micro Channel)
0096        POS 'CD SETUP' selector                             (Micro Channel)
00A0-00A1   Interrupt controller 2, 8259A                       AT              PS2
00A0-00AF   IOSGA NMI mask register                                            PS2
00B0-00BF   realtime clock/calendar, (undocumented)                            PS2
00C0-00DF   reserved                         PCjr  PC  XT  AT  CVT  M30
```