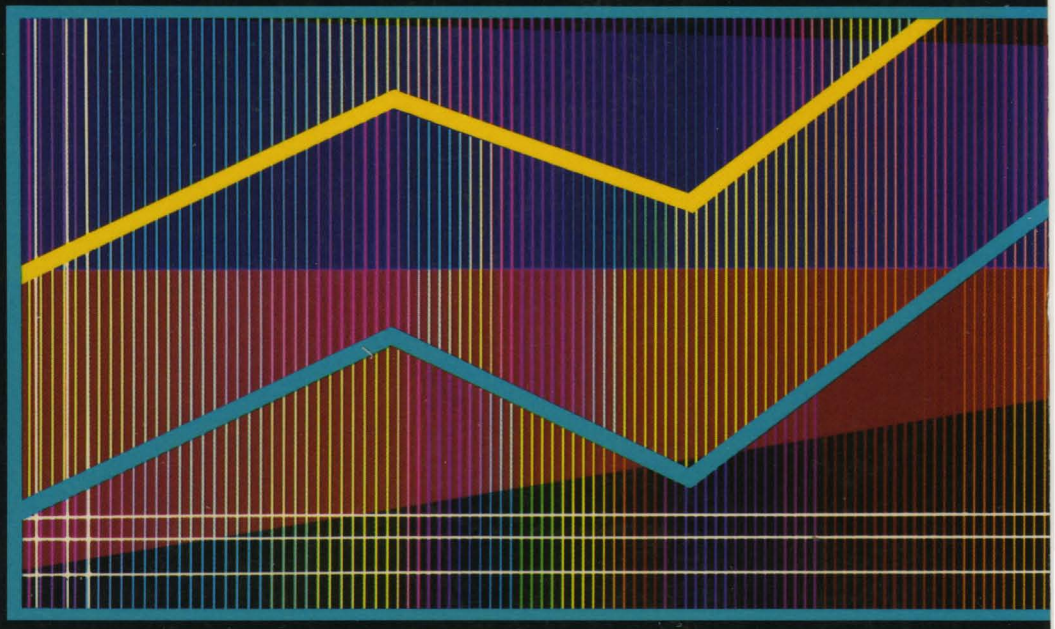
WILEY

IBM® PS/2®

USER'S REFERENCE MANUAL



GILBERT HELD

IBM® PS/2® User's Reference Manual

Related titles of interest from Wiley:

DESKTOP COMMUNICATIONS: IBM PC, PS/2 & COMPATIBLES, Honig and Hoover

IBM PS/2: A BUSINESS PERSPECTIVE, Hoskins

THE NEW DOS 4.0, Christopher, Feigenbaum, and Saliga

DOS 4.0 REFERENCE, Christopher, Feigenbaum, and Saliga

OS/2: A BUSINESS PERSPECTIVE, Conklin

OS/2: FEATURES, FUNCTIONS, AND APPLICATIONS, Krantz, Mizell, and Williams

THE PC UPGRADER'S MANUAL: HOW TO BUILD AND EXTEND YOUR SYSTEM, Held

DOS PRODUCTIVITY TIPS AND TRICKS, Held

IBM AS/400: A BUSINESS PERSPECTIVE, Hoskins

TURBO C DOS UTILITIES, Alonso

QUICK C DOS UTILITIES, Alonso

HARD DISK SMARTS: EVERYTHING YOU NEED TO CHOOSE AND USE YOUR HARD DISK, Bosshardt

LOCAL AREA NETWORKS: THE SECOND GENERATION, Madron

LANS: APPLICATIONS OF IEEE/ANSI STANDARDS, Madron

THE 80386/387 ARCHITECTURE, Morse, Isaacson, and Albert

THE 80286 ARCHITECTURE, Morse and Albert

IBM® PS/2® USER'S REFERENCE MANUAL

Gilbert Held

John Wiley & Sons, Inc.
New York Chichester Brisbane Toronto Singapore

Publisher: Stephen Kippur
Editor: Therese A. Zak
Managing Editor: Corinne McCormick
Editing, Design, and Production: Impressions, Inc.

Library of Congress Cataloging-in-Publication Data

Held, Gilbert, 1943–
IBM PS/2 user's reference manual / Gilbert Held.
p. cm.
Bibliography: p.
ISBN 0-471-62150-1
1. IBM Personal System/2 (Computer system) I. Title.
QA76.8.I25963H45 1989
005.4'469—dc20 89-34030
CIP

Printed in the United States of America

89 90 10 9 8 7 6 5 4 3 2 1

Preface

Although the first microprocessor was developed in 1969, it wasn't until 1975 that a personal computer could be purchased in kit form in the United States. From 1975 through 1981 a large number of vendors entered the personal computer field, manufacturing over a hundred different types of computers. Unfortunately, the most common characteristic among personal computers was their incompatibility with one another.

In what some people would call typical IBM product development strategy the emerging personal computer market was noted and the company waited to enter this computational area until a large enough demand developed to make a venture financially successful. In 1980, IBM formed an internal entrepreneurial group whose charter was to develop a personal computer that could be used in business or home environment and to have the product ready to sell within one year. The result was a new era in personal computing, that began with the introduction of the IBM PC in August 1981.

From a technical perspective the IBM PC was quite similar to several personal computers that were already marketed. Where the IBM PC broke new ground was in its legitimization of personal computing in the eyes of corporate America and in its open architecture. By designing the IBM PC with expansion slots and publishing the technical specification governing the slot interface, IBM made it extremely easy for third-party vendors to design compatible equipment whose installation increased the functionality of the computer.

The success of the IBM PC resulted in its acceptance by industry and the business world as a de facto standard for personal computing. Due to this acceptance, many persons questioned the necessity for what is essentially a second standard represented by a second series of personal computers that IBM introduced in April 1987, when the PS/2 family was announced.

Members of the PS/2 family differ from the original PC series in several areas, including the type of data storage media used, expansion slots supported, and video standard capability. Although these differences are not trivial, it is important to recognize the fact that both generations of personal computers support DOS. Due to this, over 99 percent of software developed for use on the IBM PC series will operate on PS/2 computers.

Once you see the operation of software on a PS/2, the answer to questions concerning the rationale for a second generation of personal computers becomes evident. The PS/2's use of 3½-inch diskettes with rigid outer cases significantly reduces the potential for data corruption due to disk damage that occurs with 5¼-inch floppy diskettes used on the original PC series. The new video standards incorporated into the PS/2 computers

provide both enhanced clarity in displaying data as well as compatibility with the display modes supported by the first generation of personal computers. Although adapter boards designed for the PC series cannot be used in the Micro Channel expansion slots of PS/2 computers, this is probably a small price to pay compared with the design and capabilities incorporated in Micro Channel architecture. The Micro Channel design requires adapter boards to have better grounding and more rigid construction than adapter boards designed for insertion in the IBM PC series of personal computers. Concerning capabilities, the Micro Channel design permits adapter boards containing independent microprocessors to operate in conjunction with the microprocessor on the system board of the PS/2. This design can permit up to 15 additional processors to be supported by Micro Channel-based PS/2s, providing a significant expansion capability that may enable its design to support new technological development into the next century. The PS/2 provides users with both a mechanism to perform desktop processing today and the capability to take advantage of tomorrow's technological development.

Acknowledgments

The preparation of a book is a comprehensive task that requires the cooperation and assistance of many individuals and organizations, as well as one's family. Thus, I would like to take the opportunity first to thank my family for their patience and understanding for those countless hours during which I examined the operation of IBM's PS/2 computers and drafted the manuscript that resulted in this book.

From personal experience, the indifference of large organizations to authors is essentially a total misconception. I would especially like to thank International Business Machines Corporation for the staff's cooperation and assistance in providing me with hardware and software that facilitated the development of this book. To Mrs. Carol Ferrell, who has diligently prepared this book and others for me, taking my handwritten notes and converting them into a manuscript suitable for publication, a special thanks is extended. To Teri Zak, who sponsored this project, I am once again indebted for her backing.

Trademark Acknowledgments

The names of products and company names listed below, and in this work, to which reference is made in this work, may be protected by Federal, State, or Commonlaw trademark laws.

1-2-3 is a registered trademark of Lotus Development Corporation.

Accunet Editor is a registered trademark of AT&T.

Apple is a registered trademark of Apple Computer, Inc.

Ashton Tate is a registered trademark of Ashton-Tate.

AST 3700 is a registered trademark of AST Research.

Brooklyn Bridge is a registered trademark of White Crane Systems, Inc.

Compaq Deskpro 386 is a trademark of Compaq.

CP/M is a trademark of Digital Research, Inc.

CP/M-86 is a trademark of Digital Research, Inc.

Dataphone Editor is a registered trademark of AT&T.

DESQview is a trademark of Disk Quarterdeck Systems, Inc.

Epson AQ850 is a trademark of Epson Corporation.

Epson AQ1050 is a trademark of Epson Corporation.

Epson is a registered trademark of Epson Corporation.

GEM is a trademark of Digital Research, Inc.

Hercules is a trademark of Hercules.

IBM is a registered trademark of the International Business Machines Corporation.

IBM PC is a registered trademark of the International Business Machines Corporation.

IRMA is a registered trademark of DCA.

IRMAcom/3770 is a registered trademark of DCA.

IRMAlink is a registered trademark of DCA.

Kermit is copyrighted by the Trustees of Columbia University. The name "Kermit" is used with permission of Henson Associates, Inc.

Lisa is a trademark of Apple Computer, Inc.

Lotus is a registered trademark of Lotus Development Corporation.

Macintosh II is a registered trademark of Apple Computer, Inc.

Macintosh is a registered trademark of Apple Computer, Inc.

Micro Channel is a registered trademark of the International Business Machines Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

Microsoft Windows is a registered trademark of Microsoft Corporation.

Microsoft Windows/386 is a registered trademark of Microsoft Corporation.

Microsoft Word is a registered trademark of Microsoft Corporation.

Mountain Computer Series 7000 Combo System is a registered trademark of Mountain Computer, Inc.

MS-DOS is a registered trademark of Microsoft Corporation.

MultiScan is a trademark of Sony.

MultiSync is a registered trademark of NEC Corporation.

NetWare is a trademark of Novell, Inc.

OS/2 and the Operating System/2 are registered trademarks of the International Business Machines Corporation.

PC-DOS is a registered trademark of the International Business Machines Corporation.

PC-MOS/386 is a trademark of The Software Link, Inc.

Personal Computer AT and AT are registered trademarks of the International Business Machines Corporation.

Personal Computer XT and XT are trademarks of the International Business Machines Corporation.

Presentation Manager is a trademark of the International Business Machines Corporation.

ProPrinter is a registered trademark of the International Business Machines Corporation.

PS/2 and Personal System/2 are registered trademarks of the IBM Corporation.

PS/2 Model 25, PS/2 Model 30, PS/2 Model 50, PS/2 Model 50Z, PS/2 Model 55 SX, PS/2 Model 60, PS/2 Model 70, PS/2 Model P70, PS/2 Model P70 386, PS/2 Model 80 are registered trademarks of the International Business Machines Corporation.

Quadboard PS/Q is a registered trademark of Quadram.

Quadram Corporation is a registered trademark of Quadram.

QuadVGA Graphics Adapter is a trademark of Quadram.

Quietwriter III is a registered trademark of the International Business Machines Corporation.

Quietwriter is a registered trademark of the International Business Machines Corporation.

SIM3278 is a registered trademark of SIMWARE.

SIM3278/VM is a registered trademark of SIMWARE.

SuperProject is a trademark of Computer Associates International.

Teletype is a registered trademark of AT&T.

UNIX is a trademark of AT&T.

X.25 adapter board is a trademark of Western Digital and Eicon Tech Corporation.

XENIX is a registered trademark of Microsoft Corporation.

Contents

1 / Hardware Overview

The IBM Personal System/2 is a family of personal computers that represent IBM's second generation of small computer technology. Unveiled in April 1987—approximately six years after the introduction of the IBM PC—members of the PS/2 family were designed to provide users with enhanced performance while retaining a high degree of compatibility with programs developed for operation on the earlier series of personal computers.

When the IBM PS/2 family was introduced, four distinct models were announced. Each of these models—30, 50, 60, and 80—could have its level of performance equated to its model number, with the PS/2 Model 80 representing the highest level of performance. Subsequent to the April 1987 introduction, IBM added five additional models to the PS/2 family: the Models 25, 50Z, 55, 70, and P70. The PS/2 Model 25 presently represents the entry-level member of the Personal System/2 family. The PS/2 Model 50Z and Model 55 can provide a higher level of performance than the Model 60, whereas the PS/2 Model 70 can provide a higher level of performance than the Model 80. The Model P70 is IBM's first portable member of the PS/2 family.

Because everyone may experience substantial confusion in comparing the performance level of different members of the PS/2 family, this chapter first examines the basic components of each computer model. Then attention shifts to the system unit, because this component is designed to contain the microprocessor memory, circuitry, and online storage devices that govern the functionality and level of performance of each computer. Using this information as a base, this chapter then examines each member of the PS/2 series to describe its level of performance, as well as its ability to operate under the control of IBM's two major operating systems—DOS and OS/2.

Although there will probably never be a typical PS/2 system, each PS/2 will include a minimum of these major components: a keyboard unit; a system unit containing space for the installation of memory, a variety of adapter boards and storage devices; a monitor; and a printer. For all members of the PS/2 family except the Model 25 and Model P70, each of the major components is a modular, standalone device with the keyboard, monitor, and printer capable of being connected via individual cables to the system unit. Both the Model 25 and Model P70 consist of a system unit and a monitor in one common housing, eliminating the requirement to cable a monitor to the system unit of that computer.

Figure 1.1 illustrates the four members of the PS/2 family that were announced in April 1987. The PS/2 Model 30 and PS/2 Model 50 are desktop units, each having a system unit that can be placed horizontally on a desktop, with your monitor resting on

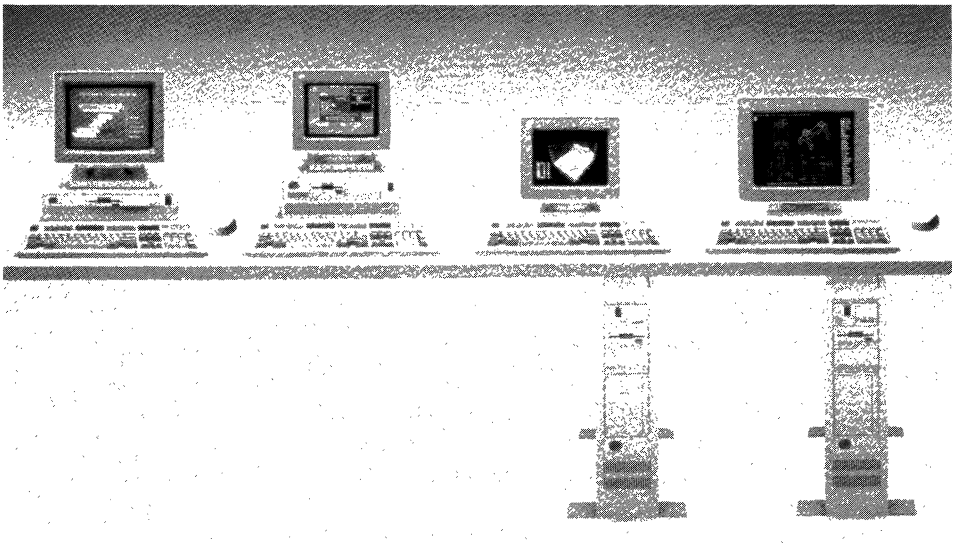


Figure 1.1
The Original Members of the PS/2 Family *(left to right)*: Models 30, 50, 60, and 80 (Photograph courtesy of IBM Corporation)

it. The PS/2 Model 60 and Model 80 have a vertically constructed system unit designed to be placed on a floor. Then your keyboard and monitor can be placed on a desktop and cabled to the floor-standing system unit.

Although each member of the IBM PS/2 family has a keyboard and system unit, the wide variety of equipment marketed by IBM and other manufacturers may mean that your system differs slightly from any of those shown in Figure 1.1. Among the major differences that can occur are the type of monitor used and the number and type of on-line storage devices that are either housed in your system unit or cabled to that component. Remaining portions of this section examine two components that are common to all PS/2s—the keyboard and system unit—although the system unit varies among the computer models.

Keyboard

Each member of the IBM PS/2 family uses the Enhanced PC Keyboard, illustrated in Figure 1.2. This 101-key keyboard is attached via a coiled cable to the rear of the system unit of your computer. The keyboard contains all of the keys found on a conventional typewriter as well as many special keys. These special keys are designed to assist the computer user in performing such tasks as programming, editing, updating, and executing programs. As an option, you can obtain an 84-key Space-Saving Keyboard for the Model 25. This keyboard is essentially the same as the Enhanced Keyboard, except that it does not have the numeric keypad.

At the top of the keyboard are 12 function keys, labeled F1 through F12, arranged in a row of three areas, with each area containing 4 keys. These keys can initiate special functions, such as displaying a menu or help information. Because each function key transmits a unique (but nonprintable) code when pressed, application programs can

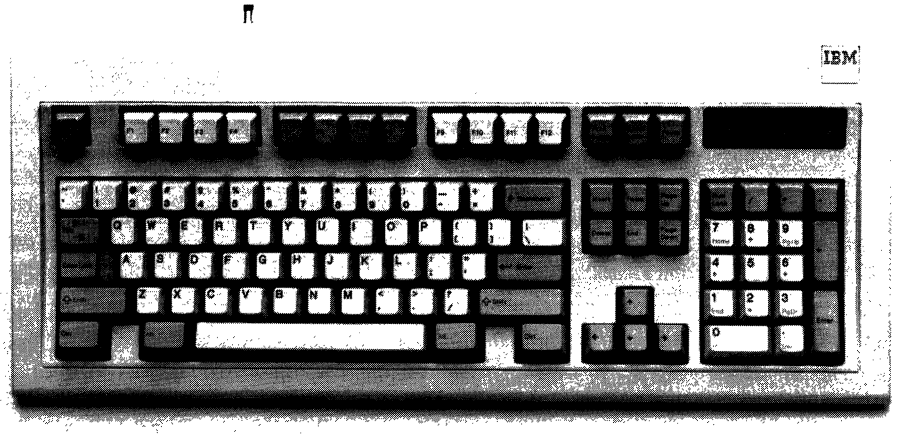


Figure 1.2
IBM Enhanced Keyboard for Any Model of PS/2 (Photograph courtesy of IBM Corporation)

Table 1.1
Initial Function Key Assignments in BASIC

Key	Command	Key	Command
F1	LIST	F6	LPT1:
F2	RUN	F7	TRON
F3	LOAD	F8	TROF
F4	SAVE	F9	KEY
F5	CONT	F10	SCREEN

assign special meanings to these keys, such as commands to load a data file or to terminate the program and return to the operating system command level. With the help of special utility programs, you can assign your own definitions to these keys. For example, you can make a single keystroke initiate a complex command sequence that you frequently enter via the keyboard.

IBM assigned predefined functions to the function keys for editing *disk operating system (DOS)* command line entries and for generating BASIC language commands. For DOS command line editing only the first five function keys have a predefined meaning. In BASIC, 10 keys are initially assigned meanings in the form of BASIC commands that are generated when each key is pressed. You can change the meaning of one or more function keys in BASIC to correspond to a function or sequence of operations you commonly perform; hence, you can save labor using these keys, because each one can reduce a long series of keystrokes to a single keystroke. Table 1.1 lists the BASIC commands initially assigned to each function key.

Most application programs—word processors, spreadsheets, and database managers—take advantage of the function keys by assigning a program function or operation to each key. As an example, pressing F2 when you are using a word processing program might invoke a spelling checker, and F5 could center the data on the line where the cursor currently resides. Note that function keys may perform different tasks for dif-

ferent application programs, because there are no standards governing the program function assigned to each key.

The numeric keypad is situated on the right-hand side of the keyboard. The keypad is useful for the rapid entry of numeric data, as on a calculator. Because most keys in this area perform dual actions—numeric values and cursor movement—you must enable their numeric usage by pressing the Num Lock key, which is above the 7/Home key. Note that in the upper right corner three indicators (labeled Num Lock, Caps Lock, and Scroll Lock) are illuminated when their respective keyboard states are enabled.

In the upper left portion of the keyboard, to the left of the F1 key, is the Esc (escape) key. As its name implies, the code generated by this key is typically interpreted by application programs as a request to escape from a current activity.

Although the keyboard may not appear to be a sophisticated device, it in fact contains an Intel 8048 microcontroller that provides a significant degree of intelligence. The microcontroller supervises all keystrokes, generates a unique code for each key, and transmits these codes to the microprocessor located inside the system unit to which the keyboard is cabled. Other tasks performed by the 8048 include a diagnostic test of the keyboard when power is applied to the system unit; preventing one keystroke from being interpreted as two, which is more formally known as *debouncing;* and checking the keyboard for stuck keys.

When you press any key, it generates a unique number known as its scan code. For the keys on the keyboard illustrated in Figure 1.2, the scan codes are numbered 1 through 101 to correspond to the number of keys on the keyboard. When you press a key, the 8048 transmits the scan code of the key to the system unit. Similarly, when you release the key, the 8048 transmits the key-release code to the system unit; the release code is the regular scan code of the key plus 128.

When a key is pressed, released, or repeated by holding it down, its action is stored in a 20-character buffer inside the keyboard. The keyboard generates an interrupt to the system unit, in effect requesting the servicing of the key action. In response to the interrupt, part of the operating system code reads the scan code from the keyboard and sends instructions back to the keyboard. This code is known as the *Basic Input/Output System (BIOS),* and it is contained on a *read only memory (ROM)* chip in the system unit. BIOS instructions tell the 8048 microprocessor in the keyboard to remove the key action from the keyboard's buffer.

The ROM BIOS routines in the system unit are responsible for monitoring all keyboard activity. That is, they keep track of the scan codes and release codes to determine whether you pressed a sequence of alphanumeric keys; held down one key to make it repeat the character; or held down the Alt, or Ctrl, or Shift key while you were pressing other keys in order to initiate some special function. The routines also keep track of the current status of the toggle keys (Caps Lock, Num Lock, and Scroll Lock). In the light of all this information, the BIOS routines are able to translate your keystrokes into the appropriate ASCII codes for processing purposes.

System Unit

The heart of each member of the IBM PS/2 family is its system unit. When viewed from the front, each desktop unit has the IBM logo in the upper left corner and at

least one 3½-inch diskette drive located to the right of the section containing the logo. Depending on the storage devices you obtain with your computer, you can have another 3½-inch diskette drive or a fixed disk installed in the housing area to the right of the first diskette. At the extreme right of the front of the system unit is a power-on light and the power switch. Figure 1.3 illustrates the front of the system unit of the PS/2 Model 50, highlighting its exterior parts.

In examining the interior of a PS/2's system unit, this chapter uses the Model 50 for illustrative purposes, as well as for referencing the similarities and differences between members of the PS/2 family of personal computers.

System Board

Figure 1.4 illustrates the system board of the PS/2 Model 50; the system board becomes visible if you remove the cover of the system unit and then disconnect and remove any previously installed on-line storage devices and adapter cards. Note that the right side of the system board illustrated in Figure 1.4 is installed to face the front of the system unit. Similarly, the left side of the system board containing parallel and serial ports, as well as the keyboard and pointing device connectors, is installed facing the rear of the system unit. This layout explains why you must cable your monitor, keyboard, and other peripheral devices to connectors located at the rear of your system unit.

The square and rectangular areas located in the lower right portion of Figure 1.4 are the microprocessor and optional math coprocessor, respectively.

Microprocessor

The microprocessor is the key to the data processing and computational capability of your personal computer. Depending on the PS/2 model you use, your system unit

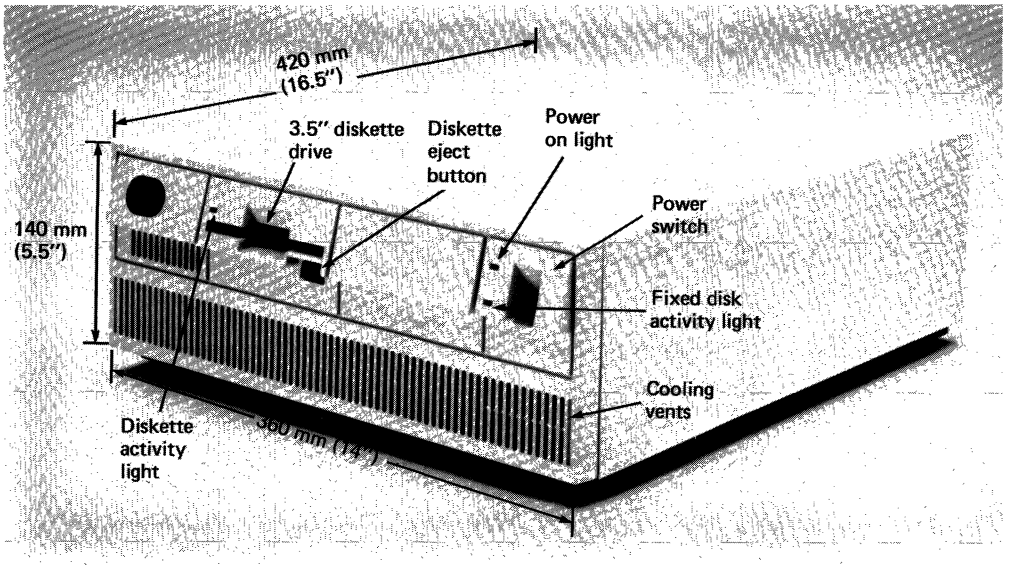


Figure 1.3
Front of the PS/2 Model 50 System Unit

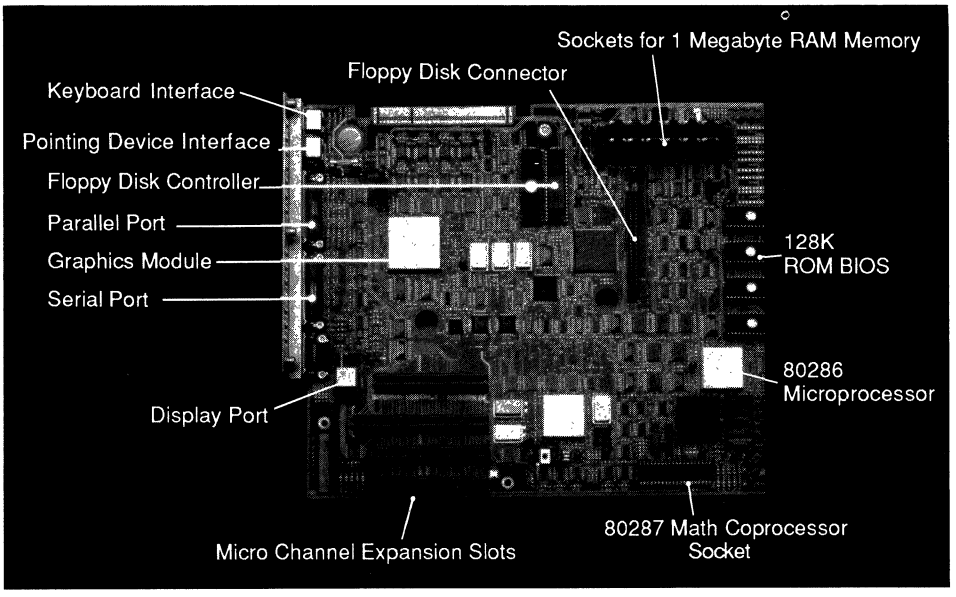


Figure 1.4
Model 50 System Board

contains one of three types of microprocessors manufactured by the Intel Corporation. Models 25 and 30 use the Intel 8086; Models 30 286, 50, and 60 use the Intel 80286; and Models 55, 70, P70, and 80 use the Intel 80386 microprocessor.

Although the PS/2 family was introduced as the successor to the IBM PC series, in actuality the microprocessors used in some PS/2 models either predate (8086) or are the same chip (80286) used in some members of the original IBM PC series of personal computers. When IBM was designing its first personal computer, Intel marketed two similar microprocessors, the 8086 and the 8088.

The Intel 8086 was first manufactured in 1978, a year before the 8088. Both microprocessors manipulate data in 16-bit increments; however, the 8086 exchanges data with memory and many peripherals in 16-bit segments, whereas the 8088 is limited by an 8-bit data bus to performing I/O operations in 8-bit segments. Even though the differences between the 8086 and 8088 are significant, their design provides program compatibility between microprocessors, permitting programs developed to operate on the 8086 with the ability to operate on the 8088. Another key difference between the 8086 and the 8088 is their operating rate.

Operating Rate Versus Clock Rate

The operating rate of a microprocessor is a function of a quartz crystal included on the system board. This crystal can be thought of as functioning similar to a metronome, providing a steady beat in increments of time that are used by different electrical elements on the system board to operate in tandem. The crystal generates pulses used to synchronize the flow of electronic pulses between the computer circuits and components in the system unit. The rate at which the crystal oscillates, or beats, is known

as its *clock rate* and governs how fast information can flow in the computer. Thus, the faster the clock, the quicker data bits can be recognized and processed.

Although it would appear that simply increasing the clock rate could make a computer more powerful, many design constraints limit the speed of the clock used in a particular computer. First, a faster clock operates at a higher frequency (more oscillations per unit of time) than a more slowly operating clock. Because high frequencies are more likely to leak than low frequencies, this means that at some crystal operating rate errors will occur due to leakage being interpreted as false pulses. A second limit to the clock rate is imposed by the memory chips used in a computer. These chips are rated for a minimum access time—that is, there is a delay between the moment when you apply the address signals and the moment the data is available on the chip output lines. If your clock rate is shorter than this delay time, you'll read incorrect data. Although both of these limitations to clock rate can be circumvented to a degree—by shielding connectors and using faster memory chips—it can be costly to do so. Thus, personal computers are designed to operate at a specific clock speed or, in some cases, at one of two clock speeds.

Clock speed is measured in megahertz (MHz), which stands for millions of cycles or pulses per second. The rate at which the crystal oscillates differs from the rate at which the microprocessor operates, because circuitry in all personal computers derives fractions of the crystal oscillation rate to operate different components mounted on the system board. Thus, one measurement of the amount of information a microprocessor can process is its operating rate and not the system clock rate. Fortunately, all personal computer manufacturers specify the actual operating rate of the microprocessor they use in their computer.

The 8086 Versus 8088 Processors

The Intel 8086 operates at 8 MHz, whereas the 8088 (used in the IBM PC) operates at 4.77 MHz. Even though the wider data path and higher operating rate of the 8086 boosts its performance to approximately twice that of an 8088, IBM selected the 8088 for its first personal computer. The reason for its selection was probably one of economics; it would have been more expensive for IBM to develop adapter cards and other parts that could take advantage of the 8086's 16-bit bus structure. Thus, the selection of the 8088 enabled IBM to take advantage of the 16-bit processing capability of the microprocessor as well as the lower costs associated with the use of an 8-bit bus architecture. Although several vendors incorporated 8086 microprocessors into their personal computers both before and after the IBM PC was introduced in 1981, it took IBM until 1987 to incorporate that chip into the PS/2 Model 25 and Model 30 computers. Even then, IBM did not take full advantage of the 8086, because the bus used in the PS/2 Model 25 and Model 30 computers was constructed using an 8-bit data path. Although this design precludes taking full advantage of the 8086, it permits adapter cards designed for use in 8088-based IBM PC and PC XT personal computers to be used in the PS/2 Model 25 and Model 30 computers.

The 80286 Processor

The Intel 80286 was first manufactured in 1982 and was used by IBM in its PC/AT, which was introduced in 1984. The 80286 is similar to the 8086 in that both are true

16-bit microprocessors, each with a 16-bit data bus. Although similar to the 8086, the 80286 has several significant differences, primarily the capabilities to operate in two different modes and to address much more memory. In addition, IBM used Intel 80286 microprocessors that operated at 6 MHz and 8 MHz in its PC/AT. In comparison, the original PC used an 8088 microprocessor that operated at 4.77 MHz, whereas the 8086 used in different PS/2s operates at 8 MHz.

Real Versus Protected Mode Operation The 80286 can be operated in either real or protected modes. When the microprocessor operates in its real mode, in essence, it functions as an 8088 or 8086 with respect to memory address capability. In this mode the 80286—like the two earlier microprocessors—can only address 1024K bytes of memory directly, because it uses a 20-bit address to access memory. The 80286 uses a 20-bit address to provide compatibility with the IBM PC and PC XT that use the 8088 microprocessor. The design of the 80286's real mode addressing is related to the design of the 8088. The Intel 8088 is a microprocessor that has 16-bit internal data paths and for each memory operation generates a 20-bit address that can access 2^{20} (1,048,576) memory locations. However, the microprocessor has only eight pins for data, so it has to perform two memory operations to fetch or store a 16-bit data word, one byte at a time. Further, because it's desirable to be able to keep program code, data, and the stack in separate blocks of memory, the microprocessor computes every memory address in two parts, known as the *segment* and the *offset.*

The segment is a contiguous area of memory, and the offset is the number of bytes from the start of the segment. Because the segment is limited to 64K bytes in length, you can represent any offset by a 16-bit number. The segment must start on a 16-byte boundary within the physical memory. The 8088 has four 16-bit segment registers, each of which points to the base of one of the four possible segments (which may overlap each other). To calculate a physical address, the 8088 multiplies the value in a segment register by 16 and then adds the logical (offset) address; the result is always a 20-bit address. Thus, the 20-bit address used by the 80286 in its real mode provides address compatibility with the 8088 and 8086 microprocessors.

In the second mode of the 80286, the protected mode, the microprocessor uses a 24-bit address bus, permitting direct addressing of 16M bytes of memory. In addition, in the protected mode of operation the microprocessor becomes capable of running several applications at one time, a process more formally known as *multitasking.*

In comparing the performance level of the 80286 with those of the 8086 and 8088, the 80286 holds over six times the number of transistors than each of the other two microprocessors. By integrating many devices onto one chip where previous processes required several chips, the 80286 can run software approximately five times faster than an 8088 when its chip design, use of a 16-bit data bus, and faster clock rate are considered.

80386

In 1985, Intel released the 80386, which is a 32-bit microprocessor. The 80386 contains approximately 275,000 transistors and can directly address 4 gigabytes of memory, which is 250 times more than the 80286 can address. In addition to processing data in 32-bit increments, the 80386 also operates at higher clock rates than the 80286, permitting software to execute three to four times faster than on an 80286-based

personal computer. Like the 80286, the 80386 can be operated in both real and protected modes. The IBM PS/2 Model 55, 70, P70, and 80 are based on the use of the 80386 microprocessor.

Coprocessor

A mathematical coprocessor similar to the Intel 80287 shown installed in Figure 1.4 is one of the few options available across the entire PS/2 product line. Each member of the PS/2 family is manufactured with a socket that accepts a particular math coprocessor that matches the microprocessor installed in the computer. The coprocessor extends the capability of the microprocessor, because it can perform certain kinds of arithmetic operations much faster than the microprocessor. The coprocessor is designed to perform floating-point arithmetic in hardware, resulting in an increase in speed, range, and precision over software floating-point arithmetic operations performed by the microprocessor.

Intel currently markets a family of math coprocessors, three of which are illustrated in Figure 1.5. Each coprocessor has a notch, which is used to align it for insertion into a socket of the motherboard of the personal computer for which it was designed.

The Intel 8087 coprocessor is used with 8086 and 8088 microprocessors. The 80287 coprocessor is used with the 80286 microprocessor, whereas the 80387 coprocessor is used with the 80386 microprocessor.

ROM BIOS

The four chips aligned vertically above and to the right of the microprocessor illustrated in Figure 1.4 are known as read only memory (ROM) chips. These ROM chips contain

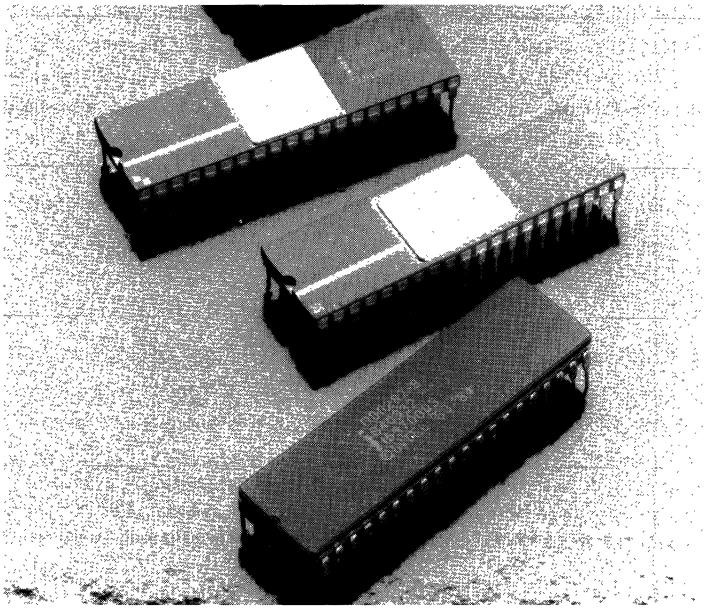


Figure 1.5
Three Intel Math Coprocessors (Photograph courtesy of Intel Corporation)

instructions or program statements that can only be read and are nonalterable—hence the name ROM. The four ROM chips illustrated in Figure 1.4 contain approximately 128K bytes of code divided into BIOS (Basic Input/Output System) and a set of BIOS routines called POST. System boards used in the Model 25 and Model 30 computers contain 64K bytes of code, similarly subdivided into different routines.

The purpose of the BIOS is to present a common interface to programs by isolating hardware usage to a set of predefined routines; such routines include displaying information on a monitor or writing data onto a file. Programmers can invoke BIOS routines using a set of interrupts routed to various BIOS entry points. This usually simplifies the programming effort because, for example, a programmer can simply load data into a register and invoke an appropriate interrupt to display information on the monitor. Without the BIOS the programmer would have to know the address of the monitor port as well as how to control it.

Programmers are encouraged to use standard BIOS calls in writing programs. Although writing directly to the hardware is faster than calling a BIOS routine, it can cause problems if a new version of the operating system uses an area in memory that a programmer previously used in developing a "custom" routine.

POST

POST, which is an acronym for Power-On Self Test, is a set of routines designed to test the keyboard, memory, and other critical elements of the computer. During POST, the computer writes data into each memory location, computing the proper parity-bit state, and compares this with the parity bit read from memory; if they differ, a parity error has occurred and the memory chip is considered to be defective. The screen displays a code in the upper left corner of the monitor to denote that a parity error occurred. The code defines the chip location that caused the error. This information can then be used to replace the failed chip. If no fatal errors are encountered during POST, a routine is then initiated to activate your computer's disk drives. This routine searches for a disk that contains the operating system and, upon finding one, loads a small program from the disk and executes it. This program, in turn, loads the remainder of the operating system. When loading of the operating system is complete, control of the computer is passed from the BIOS to the operating system. Thereafter, the BIOS functions as an intermediary through which the operating system and application programs can control system devices.

RAM Memory

Random Access Memory (RAM) is so called because you can store and retrieve data at any location in any order. In Figure 1.4, the rectangular box located at the top right of the illustration denotes the area into which RAM memory chips are inserted on an IBM PS/2 Model 50 computer. A similar location on the system board is used by other members of the PS/2 family for the installation of RAM. Because this memory is installed directly on the system board at the time of manufacture, it is also commonly referred to as on-board memory or system RAM.

Most members of the PS/2 series are manufactured with 1M byte of RAM memory installed on the system board. The two exceptions to this are the Model 25 and Model

30, which are manufactured with either 512K bytes or 640K bytes of RAM. The Model 25 is manufactured with 512K bytes of RAM; however, you can order this computer with a 128K byte expansion kit that consists of RAM chips that plug into the system board, resulting in a total of 640K bytes of memory. The Model 30 is manufactured with 640K bytes of RAM on its system board.

Types of Memory

The discussion to this point concerned only conventional memory—that is, physical memory within the contiguous 1M byte address space that can be directly addressed by an 8086 or 8088 microprocessor, or by an 80286 or 80386 microprocessor when it operates in real mode (that is, when it is emulating an 8086 or 8088).

Extended memory is the contiguous address space from 1M byte to 16M bytes. The 8086 and 8088 microprocessors, which have only 20 address lines, cannot directly address memory above 1M byte. Similarly, when operating in real mode as an 8088, the 80286 and 80386 are limited to 1M byte.

Expanded memory is any 64K byte segment of memory above the 1M byte boundary. An 8086 or 8088, or an 80286 or 80386 in real mode, can access segments of expanded memory by means of a technique known as *paging* or *bank switching.* Bank switching is the process of electronically repositioning expanded memory into the microprocessor's address range. The expanded memory is divided into 16K byte blocks called *pages,* and these are swapped into or out of an area of main storage called the *page frame.* This page frame effectively becomes a window that can look into various blocks of expanded memory, as shown in Figure 1.6.

To use expanded memory, programs must be designed to operate with a switching scheme, taking into consideration such factors as the location of the page frame and page register as well as the size of the page. Fortunately, two specifications that are basically compatible with each other have attracted the widespread support of software developers.

The first specification to gain wide support was jointly introduced by Lotus Development Corporation, Intel Corporation, and Microsoft Corporation. Known as LIM EMS, this expanded memory specification uses up to four 16K byte windows between memory locations 768K and 896K for bank switching to obtain a 64K byte window to expanded

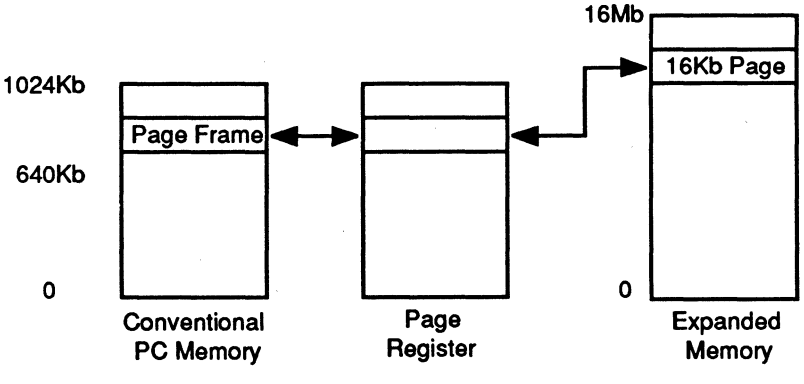


Figure 1.6
Expanded Memory Operation

memory. This window is the page frame. The LIM EMS supports up to 8M bytes of addressable memory and requires the use of a device driver to act as an interface between an applications program and expanded memory. Unfortunately, programs sold prior to the introduction of the LIM EMS, or programs not developed to use this standard, cannot take advantage of expanded memory. To do so requires one to obtain either a new release of the program that supports the expanded memory specification or a multitasking shell program under which the application program can operate.

A second extended memory specification was introduced with backing from AST Research, Quadram Corporation, and Ashton-Tate. Known as the AQA Enhanced Extended Memory Specification (AQA EEMS), this specification can be considered as a superset of LIM. The EEMS specification initially uses four page frames that are the same as the LIM specification, which makes the two compatible. Where the EEMS specification differs from LIM is in its support of up to 64 page frames, although in actuality only a subset can be used at one time due to the physical constraints of conventional memory.

Two years after the introduction of the original LIM EMS specification, it was significantly altered to increase its functionality. LIM EMS Version 4.0 incorporates such functions as multitasking and program code execution in expanded memory, equivalent to capabilities originally developed by AST for its EEMS. In fact, AST stated that there is no longer a need for EEMS as a separate specification and added its support behind LIM EMS Version 4.0. Another key feature of LIM EMS 4.0 was an expansion in maximum memory limit support, raising the amount of RAM that programs can access to 32M bytes. Thus, LIM 4.0 provides users of 8086 and 8088 based computers who require more than 640K bytes of memory an alternative to buying an 80286- or 80386-based computer system.

Selecting Memory

Before you can use expanded memory, your computer must have at least 640K bytes of conventional memory. Then you can add expanded memory cards to your computer's expansion slots to address more RAM. In fact, for 8086-based PS/2s—such as the Model 25 and Model 30, which do not support OS/2—you can only increase memory support using expanded memory.

Bus

The *bus* is the path along which signals move from the microprocessor to each of the system expansion slots that accept various types of adapter cards. The lines that are visible between system expansion slots are known as trace lines and are part of the bus. These lines provide three types of signal paths or circuits: addressing, control, and data. The control circuits direct the flow of information to and from the expansion slots; the addressing circuits enable data to reach its appropriate destination or to be read from a specific destination; the data circuits carry information to or from the CPU.

PS/2s that use the 8086 microprocessor were designed with eight data lines from the microprocessor to the system expansion slots to maintain compatibility with the IBM PC and PC XT. Thus, system expansion slots of the Model 25 and Model 30 computers can accept adapter cards manufactured for use in the PC and PC XT.

For 80286- and 80386-based PS/2 computers, IBM designed a new bus structure called the Micro Channel. The Micro Channel architecture required a complete redesign of the system expansion slots, which are now incompatible with the original PC expansion slots. As a result of this incompatibility, adapter boards that can be used in the Model 25 and Model 30 computers cannot be used in the other members of the PS/2 series and vice versa.

The Micro Channel offers several key advantages over the PC bus structure. First, and perhaps most obvious to persons familiar with the older PC series, is the elimination of DIP (dual in-line package) switches on adapter boards designed for insertion into a Micro Channel expansion slot. By contrast, the Micro Channel provides *automatic* configuration of the system and all add-in cards—a feature IBM calls *Programmable Option Select (POS).* Add-in cards designed for use in a PC bus normally require the user to set tiny DIP switches when a card is installed, as well as to set or reset one or more positions on a DIP switch housing on the system board of the computer. If you should set a DIP switch incorrectly, you might easily disable your computer. Because the POS feature permits adapter boards designed for the Micro Channel to be automatically configured when the computer is turned on and reconfigured, by software if necessary, POS simplifies the hardware installation process while eliminating the possibility of DIP switch setting errors.

The second major advantage of the Micro Channel over the PC bus is that the Micro Channel supports bus arbitration. This technique enables expansion card processors, called *masters,* to take temporary control of the computer system. Arbitration allows different devices to gain access to the system's data bus; if two or more devices attempt to control the Micro Channel at the same time, an algorithm built into hardware resolves the conflict. Through hardware arbitration, it becomes possible for the microprocessor on the system board to delegate special computing tasks to an adapter card designed specifically for that task. After delegating the task, the main microprocessor can perform other functions, resulting in a higher performance level.

The Micro Channel is designed to support a total of 16 intelligent processors, including the microprocessor on the system board. This means that the Micro Channel can support up to 15 additional processors that perform such independent functions as communications, graphics, encryption, or intelligent disk control. In fact, microprocessors included on adapter cards for installation into a Micro Channel expansion slot do not have to be compatible with the microprocessor on the system board. This is because the arbitration scheme governs access to the bus to pass data between devices, providing board manufacturers flexibility in designing their products.

Other benefits of the Micro Channel architecture include its acceptance of a new card format and pin pattern, the latter significantly reducing the possibility of the computer generating electromagnetic interference (EMI). The new card format of adapter boards designed to fit in a Micro Channel expansion slot is optimized for dense-logic design. This permits Micro Channel adapter cards to have more very large scale integration (VLSI) chips mounted on such cards than cards designed for use in an older IBM PC bus. For its pin patterns, the Micro Channel requires adapter cards to have ground pins spaced every fourth pin on both sides of the card connector. This pattern is offset by two pins on opposite sides of the connector, resulting in each signal pin

being adjacent to a radio-frequency (RF) ground. Due to this design, EMI is significantly reduced.

Disks and Storage Media

Members of the IBM PS/2 family primarily use 3½-inch diskettes and fixed disks for storing information. Each member of the PS/2 family has one or more 3½-inch diskettes and/or fixed disks installed in its system unit. To provide data-transfer compatibility with members of the IBM PC series that use 5¼-inch diskettes, IBM markets an external 5¼-inch diskette drive that can be cabled to the system unit of a PS/2.

Figure 1.7 illustrates the components of a 3½-inch diskette whose magnetic material used for recording information is encased in a shell of hard plastic to provide protection against damage. Access to the magnetic material is obtained via a sliding metal cover that is retracted only when the diskette is placed inside a disk drive. The write-protect switch located at the bottom of the diskette provides you with the ability to prevent data from being recorded onto a disk. When the switch is positioned so that the square hole in the disk is open, the diskette is write-protected. When the switch is positioned to make a visible square hole as you look down from the top of a disk, information can be written onto the diskette. Some diskettes, such as the IBM Reference Disk shipped with each PS/2 and some operating system diskettes, do not have this switch and are permanently write-protected.

Diskette Operation

The diskette is inserted into the diskette drive with the shutter on the upper shell facing toward the computer. As the diskette is inserted, its shutter retracts, providing the read/write heads of the diskette drive with access to the magnetic media.

Tracks, Bytes, and Sectors

Information in the form of data or programs is written onto and read from the diskette along concentric circles called *tracks* as illustrated in Figure 1.8. There are 80 tracks on a 3½-inch diskette, numbered from 0 to 79. Depending on the diskette used, the type of diskette drive used to format the diskette, and the personal computer user's FORMAT command specification, each track is subdivided into either 9 or 18 *sectors,* with each sector storing 512 eight-bit bytes of information.

Storage Capacity

The IBM PS/2 family supports two types of 3½-inch diskette drives—standard and high-capacity. The IBM PS/2 Model 25 and Model 30 use standard 3½-inch diskette drives. Diskettes used in these drives are formatted with 80 tracks per side, using 9 sectors per track. Thus, 2 sides times 80 tracks per side times 9 sectors per track times 512 bytes per sector yields 737,280 bytes of information, or 720K bytes of data storage capability.

Other members of the PS/2 series are manufactured with 3½-inch high-capacity diskette drives. You can use standard 3½-inch diskettes in this drive and FORMAT the

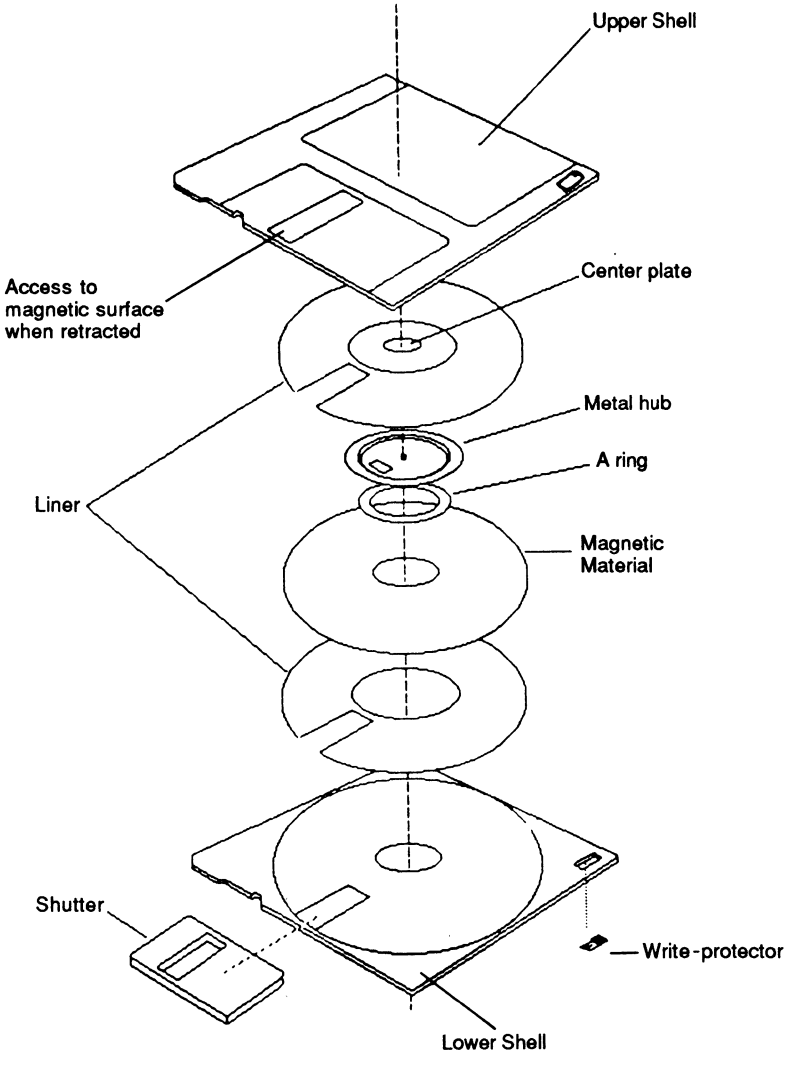


Figure 1.7
Components of a 3½-Inch Diskette

diskette to store 720K bytes of data, or you can use high-capacity diskettes to double the data-storage capacity. If you use 3½-inch high-capacity diskettes, you can format the diskette to store 18 sectors per track. The storage capacity of this diskette then becomes 2 sides times 80 tracks per side times 18 sectors per side times 512 bytes per sector, yielding 1,474,560 bytes (1.44M bytes) of information.

The Formatting Process

Before you can use a blank disk, you must *format* it; that is, you must command the computer to write dummy data to all tracks in such a way that it can later identify

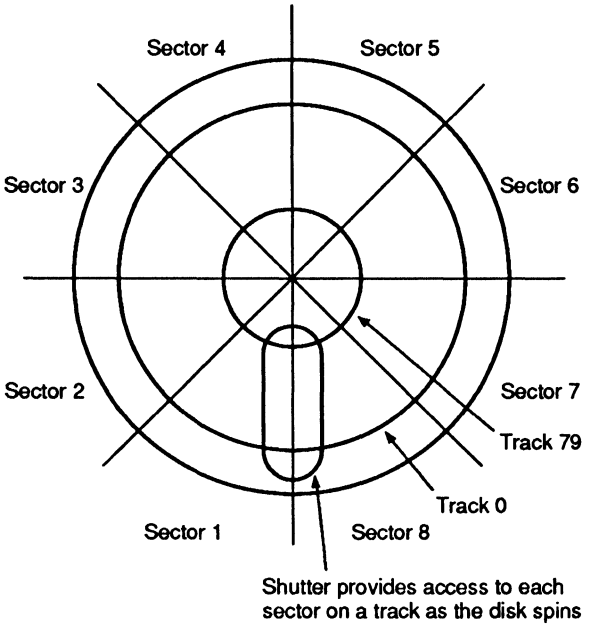


Figure 1.8
Track and Sector Relationship

precisely which part of the disk is currently under the read/write heads. During this formatting process, the computer divides the recording space into tracks, which are concentric circles numbered from 0 at the outer edge to 79 at the inner edge for a total of 80 tracks. For double-sided disks, there are two such tracks, one on each side of the disk. These upper and lower tracks are numbered identically, but are differentiated by the number of the head that reads or writes them (0 for the upper, 1 for the lower head).

The computer then divides each track into sectors, by writing a synchronization pattern consisting of the track number, head number, and sector number, followed by the appropriate number of bytes of a special character (usually E5 hex) that denotes an empty data area. After the last data byte, the computer writes two check bytes that will allow the detection of reading errors. The computer repeats this sequence of sector ID, data bytes, and check bytes as many times as there will be sectors on the track.

Fixed Disks

Fixed disk technology predates the personal computer by several decades. In 1953, legend has it that IBM's project to develop a fixed—nonremovable—disk (designated as product 3030) was "Winchester," after the .303 rifle. That name later became synonymous with the terms *fixed disk* and *hard disk.*

The fixed disks used with members of the IBM PS/2 family vary in capacity from 20M bytes to 144M bytes of storage. Unlike the PC series, which requires a disk controller to be installed in an expansion slot to control the operation of the disk, several PS/2 computers have equivalent circuitry installed in a special socket on their system

board. Those PS/2 computers need fewer expansion slots than members of the PC series, and therefore occupy less desktop space.

Operation

The fixed disk operates in a very similar way to a floppy diskette drive. Inside a hermetically sealed housing are one or more platters with a *read/write head* for each surface. The read/write head is an electromagnet capable of detecting and producing a switchable magnetic field to read and write bit streams. The read/write heads are positioned by an arm from track to track, without the heads actually touching the disk surface as they float on an air cushion several millionths of an inch in height. Data is stored on each track in groups of 512 bytes known as a *sector.* Many disks are formatted for 17 sectors per track, so that each track stores 8.5K bytes.

Tracks on a fixed disk are numbered from 0 near the circumference in ascending order toward the center of the platter, the actual number of tracks depending on the storage capacity of the disk. Figure 1.9 illustrates a schematic diagram of the track layout on a fixed disk.

To increase the capacity of a fixed disk, data bits can be recorded closer together, additional platters of data storage can be added to the device, or a combination of both techniques can be employed. When multiple platters are used in a fixed disk, multiple read/write heads are used to read and record data onto the corresponding track of each surface at the same time. The assembly of vertically corresponding tracks of each surface is known as a *cylinder* and is illustrated in Figure 1.9.

During a read or write operation, the head is first moved to the appropriate track. The time required to position the head is known as the disk's *seek time*; this time varies depending on the number of tracks across that the head must be moved in order to reach the desired track. For a single track movement, a seek time of a few milliseconds may be required, whereas a movement from the outermost to the innermost track end of the disk could require 100 milliseconds or more. The average time to position the read/write head across one-third of the disk to a random sector is known as the average access time and is usually published by disk manufacturers. Other times published by some manufacturers include track-to-track access and random reads based on defined seek widths. Once the read/write head is positioned on the appropriate track, another delay occurs until data can be read from or written onto the disk. This time is known as the *rotation time* and is the delay until the platter rotates to position the first of the sectors to be accessed under the read/write head. For a disk spinning at 3600 rpm, the average rotation time is 8 milliseconds.

Interleave Factor

The fixed disk normally rotates at 3600 rpm (or some rate between 2400 and 3600 rpm). As the disk rotates, the first sector to be read or recorded onto passes under the head, and the data transfer begins. The gap between sectors is small, so the next sector is reached very quickly—too quickly, in fact, for most disk controllers, because they usually require some time to get ready for the transfer to or from the next sector. If the controller is not ready when the second sector passes under the head, the controller must wait for an entire revolution of the disk for the appropriate sector to be correctly positioned. To eliminate this waste of time, most disks are formatted to separate logically

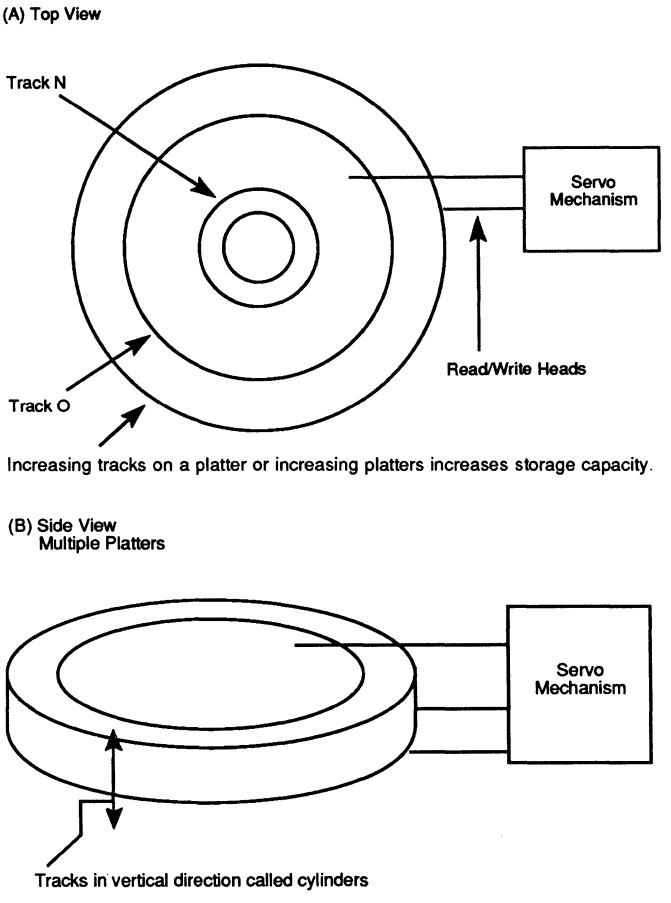


Figure 1.9 Fixed Disk Layout and Platter Operation

consecutive sectors by one or more physical sectors. This separation is called *interleaving* and provides the controller with additional time to read or record data onto consecutive logical sectors without requiring additional disk rotations.

Figure 1.10 shows two single platter disks, one formatted without sector interleaving, and the second formatted with an interleave factor of three—meaning that logically consecutive sectors are separated by two physical sectors. Normally, the interleave factor can be considered as an important parameter in comparing the performance of two fixed disk drives. As an example of the differences between interleave factors consider two disks, one with an interleave factor of two and another with an interleave factor of four. To retrieve all of the data on a track requires two disk revolutions when the interleave factor is two, but four disk revolutions if the interleave factor is four. Thus, a fixed disk with a low interleave factor is normally more efficient in storing and retrieving data than a disk with a higher interleave factor. Because the controller governs the interleave factor, you want to use the lowest interleave factor that the controller supports.

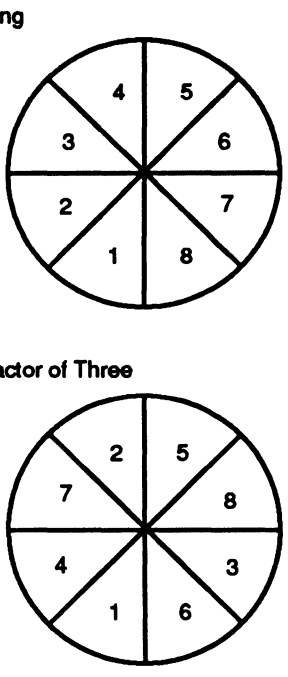


Figure 1.10
Sector Interleaving

Drive Motors

A fixed disk drive may use either a stepper motor or a voice coil motor to position the read/write heads. A stepper motor moves the heads a fixed distance and relies on the mechanical accuracy of the motor to position the heads to the correct location. Because the motor cannot adjust for media expansion or contraction of the medium, there are constraints upon how closely tracks of information may be spaced in order to be recorded and retrieved correctly. These constraints limit the storage capacity of a stepper motor system.

In a fixed disk that uses a voice coil, the motor controls the read/write head movement electronically, based on reference data stored on the disk surface. The capability to alter the reference data enables a greater number of tracks to be formatted onto a disk. In addition, many voice coil drives are so designed that when primary power is lost or turned off, the remaining power stored in the power supply's capacitors is applied to the voice coil in such a way as to retract the heads quickly to a parking position outside the data storage area of the platter.

Stepper motor drives are normally two to three times slower than voice coil drives. In addition, stepper motor drives usually have less storage capacity than voice coil drives; however, they are also less expensive. If the user does not require a large capacity drive with fast access time, such as might be necessary for a personal computer functioning as a local area network server, a stepper motor drive will normally suffice. If large-capacity storage and quick access are of primary concern, the voice coil motor drive should be considered.

Cluster

Although it is a logical rather than a physical parameter and is controlled by software, the concept of the cluster is important to understanding the efficiency of fixed disk operations. The *cluster* is the smallest addressable unit of storage space in DOS. This unit of storage corresponds to an entry in the DOS File Allocation Table (FAT), which can be considered as a map of available space on a storage medium.

Under DOS 2.X, the FAT is 12 bits in length, permitting 4096 (2^{12}) unique numbers. If each number denoted a 512-byte sector, the maximum capacity of the disk would be 4096 × 512 = 2,097,152 bytes. To increase the available storage, DOS 2.X assigns each FAT number to a cluster of 8 logically consecutive sectors. There's a trade-off, of course. The total storage on the disk is now 4096 × 4096 = 16,777,216 bytes, but the smallest amount of storage that you can allocate to a file is 4096 bytes. Thus, saving a file that contains only one character results in wasting 4095 bytes of disk storage. You can, of course, read or write a single 512-byte sector because each sector is physically identified on the disk, and the operating system can translate a logical sector number within a file to a physical track and sector number that the disk controller hardware can use.

Under DOS 3.X the FAT was changed so it could be either 12 or 16 bits in length. When 16 bits are used, the number of clusters supported increases to 65,535, permitting a cluster size of 4 sectors, or 2048 bytes, to be used. Thus, fixed disks formatted under DOS 3.X store data more efficiently than fixed disks formatted under DOS 2.X.

Landing Zone

To prevent data from being damaged due to vibrations or a bump to the computer system, many manufacturers incorporate what is known as a *landing zone* into their drives. The landing zone is a fixed location that does not contain data, onto which the read/write heads are positioned whenever power to the system turns off or is lost. Although the use of a landing zone alleviates potential damage to data due to vibrations or small bumps, if the PS/2 or PC is to be moved you should first use the IBM Diagnostic Diskette and select the `Preparing to Move` option. This option parks the heads into a fixed position and is a much better protection mechanism than a reliance on the automatic landing zone.

Video Display Support

Unlike the earlier PC series in which video display support is in the form of adapter boards installed in expansion slots, members of the PS/2 family have built-in graphics chips on the system board of each computer. The graphics chips used in the PS/2 family of computers provide two new standards of video display capability over the original PC series as well as downward video standard compatibility with the video standards supported by IBM's first generation of personal computers.

The IBM PS/2 Model 25 and Model 30 include a built-in graphics chip that provides Multicolor Graphics Array (MCGA) display support. MCGA supports the 320 by 200 pixel graphics resolution of the IBM Color Graphics Adapter (CGA), which was IBM's first product to display color and graphics, enabling up to four colors to be simultaneously

displayed from a palette of 16 colors. In addition to CGA support, MCGA adds three additional video modes—320 by 200 pixels with up to 256 simultaneous colors from a palette of 256,000 colors; a 640 by 480 pixel mode in two colors; and a monochrome mode that supports up to 64 simultaneous shades of gray.

The PS/2 Model 30 286, the PS/2 Model 50, and all higher models use the Video Graphics Array (VGA) chip, which is a superset of the MCGA. The VGA adds a 640 by 480 pixel mode that can display 16 colors simultaneously from a palette of 256,000 colors.

The VGA standard incorporates 17 different video modes on a single chip. To obtain an appreciation for the capabilities of the VGA and MCGA chips, a review of the earlier video standards is warranted.

PC Video Standards

When the IBM PC was introduced in 1981, users could select from two different video standards—Monochrome Display Adapter (MDA) and Color Graphics Adapter (CGA). The MDA standard resulted in a 720 by 350 pixel display capability in which characters are formed in a 7 by 9 pixel matrix within a 9 by 14 pixel box. Unfortunately, IBM's MDA provides neither pixel-addressable graphics nor color.

The IBM CGA standard provides both pixel-addressable graphics and color display capability. The CGA standard includes two display modes—320 by 200 pixels, in which 4 colors from a palette of 16 can be displayed, and 640 by 200 pixels in monochrome.

Users who selected the color graphics adapter to drive a color display obtained a coarser and less pleasing text display capability because characters were displayed in a 7 by 7 pixel matrix in an 8 by 8 box, in comparison to the 7 by 9 matrix within a 9 by 14 box of the monochrome display; however, they also obtained bit-mapped graphics capability. Because the use of two display adapters and two displays was expensive and wasted a valuable system expansion slot, it was not long until both IBM and third-party vendors introduced monochrome graphic display adapters that could be used with the IBM Monochrome Display.

Shortly after the IBM PC was introduced, Hercules Computer Technology introduced a video board that was IBM MDA compatible and that added a graphics resolution of 720 by 348 pixels. The Hercules graphics card's success resulted in its functionality being cloned by many manufacturers of video adapters and its graphics mode being accepted as a de facto standard.

With the growth of computer assisted design (CAD), page publishing, and other graphics intensive applications a driving force, IBM developed a new video standard known as the Enhanced Graphics Adapter (EGA). The resulting EGA video mode provided higher resolution graphics and the display of more colors than the CGA as well as better looking text characters. In its text modes, characters formed in an 8 by 14 pixel box are nearly as readable as in the MDA standard. Graphics modes permit up to 640 by 350 pixel-addressable graphics, with the display of up to 16 colors from a 64 color palette. In addition to being backward-compatible with the CGA standard, the EGA can also be used to generate 640 by 350 pixel monochrome graphics, a mode similar to the Hercules Graphics Card.

Although the EGA standard offered significant improvements over prior video resolution and color capabilities, it was still limited with respect to evolving applications.

Many third-party manufacturers began to produce EGA+ cards that offered higher resolution and IBM responded to a loss of market share by introducing its MCGA and VGA standards as chips installed on the system board of PS/2 computers.

For PS/2 users not satisfied with the MCGA or VGA, several upgrade possibilities exist. For MCGA computers, you can upgrade to VGA capability by the installation of an adapter card in the system unit of a Model 25 or Model 30. For PS/2 computers that support the Micro Channel, you can install IBM's 8514/A display adapter. This adapter must be used in conjunction with the firm's 8514 color display to achieve its maximum resolution capability.

Monitors

Unlike IBM's previous monitors that were strictly digital, only analog monitors can be used with members of the PS/2 family. The older digital monitors were controlled by four or six lines that were used by the adapter board in the system unit to inform the monitor which color to display. Thus, CGA monitors with four lines were limited to 16 (2^4) total colors, while EGA monitors with six lines were limited to a total of 64 (2^6) colors. In comparison, the video chips for the MCGA and VGA standards describe the amount of color by varying the voltage on an analog line between 0V and 1V, with a higher voltage resulting in a brighter color. The VGA monitor uses three analog inputs, one for each of the red, green, and blue primary colors. By having the ability to generate 64 values for each of the primary colors, a total of 262,144 ($64 \times 64 \times 64$) colors becomes available for display, of which 256 can be displayed simultaneously.

Monitor Selection Considerations

In selecting an appropriate monitor for your PS/2 you should examine its dot pitch, horizontal and vertical scan frequency, and adapter or video chip compatibility. The dot pitch is measured in millimeters and indicates the spacing of the pixels from one another. In general, the smaller the dot pitch, the sharper the characters will appear on a display.

The display capability of a monitor results from an electron beam that is passed from right to left over each line of the screen. This beam's intensity is varied in proportion to the intensity of the image to be displayed on the screen; a memory buffer in the video display adapter or video chip stores a bit pattern of the image to be displayed. Thus, the data in the buffer tells the monitor when to turn the electron beam on or off.

The rate at which the video adapter reads the contents of the display buffer and translates them into a screen display corresponds to the scan rate of the adapter. This scan rate actually consists of horizontal and vertical scan components. The rate at which the electron beam sweeps from right to left across the screen is known as the *horizontal scan frequency,* whereas the rate at which the beam moves from the top to the bottom of the screen is known as its *vertical scan rate.* In general, the faster the scan rate the lower the amount of screen flicker, because a high scan rate allows a screen to be redrawn in a shorter amount of time.

The VGA's horizontal scanning frequency is 31.5 KHz, which is approximately double the CGA's 15.75 KHz rate, almost double the MDA's 18.4 KHz rate, and almost 50 percent above the EGA's 21.85 KHz rate. The VGA's vertical scan rate is 70 Hz, in

comparison to the 60 Hz rate of the CGA and EGA adapters and the 50 Hz rate of the MDA adapter.

PS/2 computers operate only with analog monitors, because of the design of the graphics chips used in each computer. Because the video adapters used with the PC series support only digital displays, you cannot normally use monitors designed for the MDA, CGA, or EGA video modes with a PS/2 computer. The primary exception to this is multiscan monitors that can operate up to the 70 Hz rate of the VGA adapter and that can be reset to support analog input.

IBM Monitors

IBM offers four types of monitors for use with its PS/2 series of computers. All four monitors share a number of attributes, including the ability to accept analog input at the scan rate associated with VGA.

8503 Monochrome Display

The IBM 8503 Monochrome Display has a 12-inch diagonal screen. This display is capable of generating up to 64 shades of gray on a paper-white background. The 8503 weighs 19 pounds and is encased in a pearl-white housing that matches the color of the PS/2's system unit. The monitor is approximately 12 inches square, resulting in a 144-square-inch footprint, and it supports a maximum resolution of 640 by 480 pixels.

8512 Color Display

The IBM 8512 Color Display has a 14-inch diagonal screen. This display weighs approximately 33 pounds to include a tilt-and-swivel base. Like the 8503 monochrome display, the maximum resolution of the 8512 is 640 by 480 pixels. The 8512 has a coarse dot pitch of 0.41 mm that produces only marginally legible characters. Because of this, the 8512 should probably be restricted to moderate daily use.

8513 Color Display

The IBM 8513 Color Display uses a 0.28 mm dot pitch, creating a clearer image than that obtainable from the 8512. The 8513 has the same maximum resolution as the 8503 and 8512 monitors; however, because of its better dot pitch than the 8512 for a retail-price difference of less than $100, the 8513 is more suitable for frequent usage.

8514 Color Display

At the top of the line of IBM monitors is the firm's 40-pound 8514. Unlike the other IBM monitors, which are compatible only with the MCGA and VGA scan rates, the 8514 is a multiscan monitor that you can use with MDA, CGA, and EGA adapters in addition to the PS/2 video adapters. The maximum resolution of the 8514 is 1024 by 768 pixels; however, this resolution is only obtainable when the IBM 8514/A Display Adapter is installed in a Micro Channel expansion slot on a PS/2 Model 50 or higher computer.

Having examined the major components of the PS/2 computer family, you are now ready to examine each computer in detail.

In-Depth Look at PS/2 Models

This section describes the unique features of each PS/2 model currently marketed and compares each model to others in the family.

Model 25

The PS/2 Model 25 illustrated in Figure 1.11 represents an entry-level computing capability whose marketing target is the educational and home user. This computer, like the Model 30 that was actually introduced approximately six months before the Model 25, uses the 16-bit Intel 8086 microprocessor with an 8-bit bus design operating at 8 MHz. Unlike other members of the PS/2 family that have separate system units and monitors, the Model 25 combines both into one compact housing.

The Model 25 can be obtained in several configurations. A basic Model 25 comes with one 720K byte 3½-inch diskette drive, as well as a disk controller, serial and parallel ports, and an MCGA video controller mounted on the system board. The computer comes with 512K bytes of memory that can be expanded to 640K bytes.

Both the Model 25 and Model 30 are manufactured with IBM PC type expansion slots. The Model 25 contains two PC type expansion slots—one for a full board and one for a three-quarter length card that can be up to eight inches in length. Included

Figure 1.11
The PS/2 Model 25 (Photograph courtesy of IBM Corporation)

on the system board is a pointing device (alias a mouse), port, and socket for an optional 8087 math coprocessor.

Initially, the Model 25 was available in two configurations. The Model 25-001 has a 12-inch analog monochrome monitor, whereas the Model 25-004 incorporates a 12-inch analog color monitor in its housing. In 1988, IBM added a new Model 25 with a Token Ring Network Adapter for use on local area networks. Similar to the initially announced Model 25, the Model 25LS, which includes a Token Ring Network Adapter, can be obtained with a monochrome or a color display.

Both the Model 25 and Model 25LS support either an optional second 3½-inch diskette or a 20M byte fixed disk. The key differences between the Model 25 and Model 25LS are in the area of standard memory and available expansion slots. The Model 25LS comes with 640K bytes of RAM, while the Model 25 has 512K bytes, expandable to 640K bytes. Because the Token Ring Network Adapter uses one expansion slot, the Model 25LS only has one free expansion slot. In comparison, the Model 25 has two free expansion slots.

Two keyboards are offered for both versions of the Model 25. A space-saving 84-key model, which has a cursor control pad but no numeric keypad, can be obtained. As an alternative, the enhanced 101-key keyboard that is standard with other members of the PS/2 series can be obtained for use with the Model 25.

Model 30

Similar to the Model 25, the Model 30 is a desktop computer system that uses the Intel 8086 16-bit microprocessor with an 8-bit bus. The Model 30 can accept adapter cards manufactured for use in the original IBM PC series. Unlike the Model 25, which incorporates the system unit and monitor in one housing, the Model 30 was designed so that its system unit can be cabled to any analog monitor; thus, purchasers have the ability to turn to third-party vendors to satisfy their monitor requirements.

When announced by IBM, the Model 30 was marketed in two configurations. The Model 30-002 includes two 720K byte 3½-inch diskette drives; the Model 30-021 has one 720K byte diskette drive and one 20M byte fixed disk. Both configurations come with 640K of RAM, which, according to IBM, is the maximum system memory supported by the Model 30. Memory on each Model 30 configuration consists of 128K bytes of RAM soldered to the system board and two 256K byte banks of IBM's Single Inline Package (SIP) memory modules that can slide out. Although IBM does not currently offer memory beyond 640K, RAM expansion is possible by removing the SIP memory and installing third-party memory.

The key to the functionality of the Model 30, as well as other members of the PS/2 family, is the system unit of the computer. Both versions of the Model 30 include built-in serial, parallel, and pointing (mouse) device ports as well as built-in MCGA graphics and a built-in diskette controller. Because so much functionality was added to the system board only three expansion slots were included in the Model 30. These expansion slots are placed horizontally, parallel to the system board to save space, permitting the system unit's footprint to be reduced to 16 by 15.6 inches. The expansion slots are formed on the Model 30 by the use of a card that contains three sockets and that is attached to the system board at a 90 degree angle. This card is held in an upright position by a

plastic bracket. To install an expansion board you must first pry off a strip of plastic on the rear panel of the system unit, enabling the board to be slid into the expansion bus connector card.

Figure 1.12 illustrates the back panel of the system unit of the Model 30, illustrating the relationship between the connectors on the system unit and the location of the three horizontal expansion slots.

Like all PS/2 computers, the appearance of the Model 30 system unit (shown at the left of Figure 1.1) differs in two respects from the system unit of the members of the original PC series. First, and probably most practical, is the placement of the power switch on the front right corner of the system unit. Second, because the device housing areas for 3½-inch storage devices are smaller, you can't install 5¼-inch drives in the system unit of any member of the Personal System/2 computer family. Users who need the use of 5¼-inch diskettes can obtain an external diskette drive that is cabled to the rear of the system unit.

Limitations of Models 25 and 30

Although considered by IBM to be a member of the PS/2 family, in actuality Models 25 and 30 can be considered to be enhanced members of the PC series. The use of the 8086 microprocessor, which does not support protected-memory operations, precludes running OS/2 on either model. In addition, the use of an 8-bit PC expansion slot architecture in the Model 25 and Model 30 instead of the Micro Channel bus precludes the development of adapter boards with separate processors that can communicate using a bus arbitration scheme, as is possible with the PS/2 Model 30 286 and PS/2 Models 50 and above. Despite these limitations, the 8 MHz operating rate of the 8086 boosts the overall performance of the Model 30 to more than twice that of the 8088-based IBM PC and PC XT for performing computations.

Other standard features of the Model 30 include 640K bytes of memory, 70-watt power supply, and battery-powered clock. The dimensions of the system unit are 16

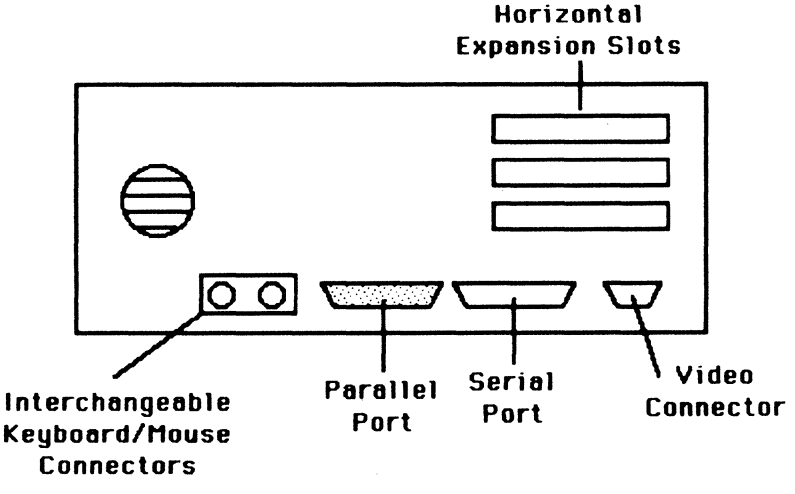


Figure 1.12
PS/2 Model 30 System Unit Back Panel

inches wide by 4 inches high, and 15.6 inches deep; thus, the desktop footprint is approximately 25 percent smaller than that of a PC or PC XT.

Model 30 286

In late 1988, IBM supplemented its PS/2 product line with the addition of two 80286 microprocessor based computers. Both Model 30s include three dual-8 and 16-bit bus PC/AT style system expansion slots in place of the 8-bit bus expansion slots built into the Model 30s. This enables each Model 30 286 to use the large number of 8- and 16-bit adapter cards originally designed for use in the PC series to include the PC, PC XT, and PC/AT personal computers.

Like the original Model 30s, each Model 30 286 includes many standard features built into its system board. These standard features include a diskette controller, serial and parallel ports, mouse and keyboard ports, and time-of-day clock. The Intel 80286 microprocessor operates at 10 MHz, providing a system throughput approximately twice that of the Model 30.

The Model 30 286 is marketed in two versions: one with a 20M byte fixed disk drive and one without. Both versions include one high-density 3½-inch 1.44M byte diskette drive and can support up to 4M bytes of memory on the system board and up to 16M byte of total system memory.

Standard features of the Model 30 286-E01 include 512K bytes of memory, expandable to 1M byte, 2M bytes, or 4M bytes on the system board; one 3½-inch 1.44M byte diskette drive; VGA display capability slots that accept most IBM PC, PC XT, and PC/AT adapter cards. The Model 30 286-E21 adds a 20M byte fixed disk drive with integrated controller to the standard features previously described.

The Video Graphics Array (VGA) chip used by the Model 30 286 is a superset of the MCGA. The VGA adds a 640 by 480 pixel mode that can display 16 colors simultaneously.

Model 50

The Model 50 illustrated as the second from the left in Figure 1.1 is a desktop system. This model has only one configuration—the 50-021—which includes one 1.44M byte, 3½-inch floppy diskette drive, and a 20M byte fixed disk. The floppy disk drive, as you shall see later, can have serious compatibility problems, both with the PS/2 Model 25 and Model 30 and with members of the original PC family that use this type of drive.

The Model 50 comes with 1M bytes of RAM, and you can expand the memory to 8M bytes. The microprocessor is an 80286, which has a 16-bit external data path. For the sake of upward compatibility, the Model 50 uses a 16-bit subset of IBM's 32-bit Micro Channel. The Micro Channel bus is physically and electrically incompatible with that of the IBM PC series, so you can't use 8-bit adapter cards designed for the PC series in any PS/2 computer except the Model 25 and Model 30. Like the Model 30 286, the Model 50 and all higher models use the Video Graphics Array (VGA) chip, which is a superset of the MCGA. The VGA adds a 640- by 480-pixel mode that can display 16 colors simultaneously.

The Model 50's system unit contains four 16-bit Micro Channel expansion slots. Because one is used by the standard equipment hard disk controller, only three are

actually available for use. Because of the use of 3½-inch disk technology, and proprietary integrated circuit chips on the system board that replaced many separate chips on earlier computers, the Model 50's system unit was shrunk to 5½ inches high by 14 inches wide by 18½ inches deep.

Limitations of the Model 50

Although the PS/2 Model 50 is more powerful than the PC/AT it was designed to replace, its 20M byte fixed disk storage capacity, as well as the slow 80-millisecond access time, was considered by many persons to be a serious limitation. The slow access time made the Model 50 impractical as a server on a local area network as well as for use as a standalone system for disk-intensive applications. The paltry 20M bytes of on-line storage do not leave much space for application programs once OS/2 is installed. Probably due to these limitations, IBM introduced the Model 50Z.

Model 50Z

To outward appearance, the Model 50Z is the same as the Model 50. Internally, there are several significant differences between these two members of the PS/2 family.

The first significant difference between the Model 50 and the Model 50Z is in the number of wait states. The Z in "Model 50Z" denotes that this member of the PS/2 family operates using zero wait states. IBM used RAM chips designed for 85-nanosecond access in the Model 50Z, instead of the 125 ns chips used in the Model 50. This permitted IBM to redesign the system board to remove the wait state from memory reads and writes.

The second significant difference between the Model 50Z and the Model 50 is in disk drive access time and storage capacity. The Model 50Z can be obtained with either a 30M byte or a 60M byte fixed disk, both disks having a relatively fast access time of 39 ms. Other changes between the Model 50 and Model 50Z include the increased utilization of integrated chips, which reduced the size of the Model 50Z's system board, and a choice of either 1M bytes or 2M bytes of RAM on the system board of that computer, in comparison to 1M byte on the Model 50.

Model 55 SX

During May 1989, IBM extended its PS/2 family with the addition of the Model 55 SX. This personal computer is based on the use of the Intel 80386SX microprocessor, a less powerful version of the 80386. The 80386SX can perform up to approximately 40 percent faster than the fastest IBM 80286-based PS/2, although the microprocessor costs considerably less than an 80386. Thus, at its introduction many computer analysts touted both the aggressive price of the PS/2 Model 55 SX, as well as its perceived significant price/performance because it provides end-users with the power, performance, and functionality of an 80386 desktop computer at prices approaching an 80286 system.

The PS/2 Model 55 SX is housed in a system unit identical to that of a PS/2 Model 30. The Model 55 SX comes standard with one 3½-inch high-capacity diskette drive and can be obtained with either a 30M byte or 60M byte fixed disk drive. Similar to the Model 50 and Model 70, the Model 55 SX contains three Micro Channel expansion slots. The Model 55 SX includes 2M bytes of RAM on the system board and the computer

can support a total of 16M bytes. Like all members of the PS/2 family above the Model 30, the Model 55 SX comes standard with VGA graphics capability; built-in mouse; serial, parallel and keyboard ports; and a diskette controller.

Model 60

Like the Model 50, the Model 60 uses an 80286 microprocessor and the 16-bit subset of the Micro Channel bus. Unlike the smaller models, however, the Model 60 is a floor-standing system (illustrated second from the right in Figure 1.1) that comes in two configurations. The 60-041 includes a 1.44M byte, 3½-inch floppy disk drive and a 40 ms average access time, 44M byte fixed disk; the 60-071 uses the same floppy disk drive but comes with a 70M byte, 30ms access time fixed disk.

The Model 60 shares many of the basic design and performance features of the Model 50, including the 10 MHz operating rate of the 80286, 1M byte of RAM on the system board, as well as built-in serial, parallel, keyboard, and mouse connectors. Unlike a desktop system, the Model 60's system unit is constructed as a vertical tower that is 23 inches tall by 6½ inches wide by 19 inches deep. Optional mass-storage expansion includes a second 1.44M byte, 3½-inch disk drive, a second 44M byte fixed disk on the Model 60-041, and either a second 70M byte or 115M byte fixed disk on the Model 60-071.

The Model 60 has seven available expansion slots, whereas the Model 50 and Model 50Z have only three. Another significant difference between the Model 50 and Model 60 is in the amount of on-line storage. The Model 60 has the potential to support up to 185M bytes of on-line storage, whereas the Model 50 is limited to 80M bytes, and the Model 50Z to 120M bytes. The Model 60 is therefore more suitable as a file server on a local area network, because a server requires a substantial amount of shared storage capacity.

Model 70

The Model 70 is IBM's first desktop PS/2 to use the Intel 80386 microprocessor. Figure 1.13 illustrates the Model 70, which, you will note, is similar in appearance to other desktop members of the PS/2 family. Three versions of the Model 70 were introduced by IBM approximately one year after the PS/2 was announced. The chief difference between each version is in the operating rate of the 80386 microprocessor and the disk storage capacity contained in the computer's system unit.

Like earlier members of the PS/2 series, each version of the Model 70 includes built-in serial, parallel, keyboard, and mouse ports. Like Models 50, 50Z, and 60, each Model 70 comes with one 1.44M byte, 3½-inch disk drive as standard equipment.

The Model 70-E61 uses a 16 MHz Intel 80386 processor and has a 60M byte fixed disk with a 29 ms average access time. The Model 70-121 runs at 20 MHz and uses a 120M byte fixed disk that has an average access time of 23 ms. The Model 70-A21 is the fastest desktop machine marketed by IBM; its 80386 microprocessor operates at 25 MHz.

From the outside, each Model 70 appears similar to Models 50s and 50Z, occupying the same footprint of 14 by 16½ inches. Like the Model 50 and Model 50Z, each version of the Model 70 has three available Micro Channel expansion slots and 2M bytes of 80

Figure 1.13
The PS/2 Model 70, Including the Intel 80386 Processor (Photograph courtesy of IBM Corporation)

ns RAM on the system board, expandable to 8M bytes. You can add another 8M bytes of 32-bit memory for a total of 16M bytes of RAM.

Comparing the Model 60 and Model 70 is similar in many respects to comparing the Model 50Z to the Model 60. Both the Model 50Z and Model 70 operate faster than the Model 60; however, the Model 60 has seven expansion slots, whereas Models 50Z and 70 have only three. Thus, you would select a Model 50Z or Model 70 if processor speed is your primary need, Model 60 if you need expandability.

Model P70 386

The Model P70 386 is IBM's first Micro Channel portable computer. This computer is designed for the traveling executive, scientist, or business professional who requires the power, performance, and functionality of a portable personal computer that incorporates the Intel 80386 microprocessor.

One of the most distinguishing features of the P70 386 is its 10-inch gas plasma display, which provides an exceptional level of visibility for a portable computer. Un-

fortunately, the gas plasma display, diskette, and fixed disk drives and 4M bytes of standard memory contribute to the portable's hefty weight—21 pounds without its battery pack.

The PS/2 Model P70 386 is available with either a 60M byte or 120M byte fixed disk. Like the Model 55 SX, the P70 386 comes with 4M bytes of system board memory that is expandable up to 16M bytes. With the computer's built-in VGA capability, the gas plasma display provides 640 by 480 pixel graphics resolution with 16 shades of gray.

Two Micro Channel expansion slots are built into the Model P70 386. One expansion slot is half the length of a conventional slot. This half-length slot is primarily designed for an internal modem card. When the system is used in that configuration, one full-length Micro Channel expansion slot is available to support the use of other options.

Model 80

The flagship of the PS/2 family is the Model 80; the processor is an 80386 that can perform both 16- and 32-bit data transfers. Three configurations are offered: the 80-041, which runs at 16 MHz and comes with 1M byte of RAM, a 1.44M byte floppy disk drive, and a 44M byte fixed disk; the 80-071, which also runs at 16 MHz and uses the same floppy disk drive, but has 2M bytes of RAM and a 70M byte fixed disk; and the 80-111, which runs at 20 MHz and comes with 2M bytes of RAM and a 115-byte fixed disk. At the time of its introduction, the Model 80-141 cost approximately $11,000 and provided users with as much processing power as an IBM System/370 mainframe that in 1975 sold for $3.4 million.

Like the Model 60, the system unit of the Model 80 is a vertically constructed, floor-standing unit. The system board can hold up to 4M bytes of RAM and has three 32-bit and four 16-bit Micro Channel expansion slots. As in all PS/2 computers, the system board includes built-in serial and parallel ports, as well as keyboard and mouse connectors.

Table 1.2 presents a comparison of the major features of the different members of the PS/2 family.

Table 1.2 IBM PS/2 Family Comparison

	Model 25	**Model 30**	**Model 30 286**	**Model 50**	**Model 50Z**	**Model 55 SX**	**Model 60**	**Model 70**	**Model P70 386**	**Model 80**
Microprocessor	8086	8086	80286	80286	80286	80386SX	80286	80386	80386	80386
Clock Speed	8 MHz	8 MHz	10 MHz	10 MHz	10 MHz (Zero Wait State)	16 MHz	10 MHz	16, 20, 25 MHz	20 MHz	16, 20 MHz
Keyboard Keys	84,101	101	101	101	101	101	101	101	101	101
Standard Memory	512, 640 Kb	640 Kb	512 Kb	1 Mb	up to 2 Mb	4 Mb	1 Mb	up to 2 Mb	4 Mb	up to 2 Mb
Expandable to	640 Kb		16 Mb	16 Mb	16 Mb	16 Mb	16 Mb	16 Mb	16 Mb	16 Mb
Video Support	MCGA	MCGA	VGA	VGA	VGA	VGA	VGA	VGA	VGA	VGA
Diskette Size and Capacity	3½ inch 720 Kb	3½ inch 720 Kb	3½ inch 1.44 MB	3½ inch 1.44 Mb	3½ inch 1.44 Mb	3½ inch 1.44 MB	3½ inch 1.44 Mb	3½ inch 1.44 Mb	3½ inch 1.44 Mb	3½ inch 1.44 Mb
Fixed Disk[1]		20 Mb	20 Mb[6]	20 Mb	30 Mb, 60 Mb	30 Mb, 60 Mb	44 Mb, 70Mb	60 Mb, 120 Mb	60 Mb, 120 Mb	44 Mb, 70 Mb, 115 Mb, 314 Mb
Additional Options[2]	3½ inch 720 Kb drive or 20 Mb fixed disk		20 Mb for E01 version	60 Mb	60 Mb for 031 version		44 Mb, 70 Mb, 115 Mb			44 Mb, 70 Mb, 115 Mb, 314 Mb
Maximum Configuration[3]	20 Mb w/option	20 Mb	20 Mb	60 Mb	60 Mb	30 Mb, 60 Mb	185 Mb	120 Mb	60 Mb, 120 Mb	628 Mb
Expansion Slots[4]	2[5]	3	3	3	3	3	7	3	2[7]	7
Operating System(s)	DOS	DOS	DOS, OS/2	DOS, OS/2	DOS, OS/2	DOS, OS/2, AIX PS/2	DOS, OS/2	DOS, OS/2, AIX PS/2	DOS, OS/2	DOS, OS/2, AIX PS/2

Notes:
1. Model 30 comes in two diskette and 1 diskette plus fixed disk configurations.
2. Model 25 version with IBM Token Ring Network Adapter card is also available.
3. The IBM 3363 Optical Disk Drive can provide an additional 200 Mb to 1.6 GB of on-line storage, depending on the model.
4. Models 25 and 30 use IBM PC expansion slots, Model 30 286 uses IBM PC/AT expansion slots, others include Micro Channel slots.
5. One slot is 8 inches in length.
6. The Model 30 286-E21 includes a 20 Mb fixed disk. The Model 30 286-E01 is a diskette-based system.
7. One is half-length.

2 / Hardware Enhancements

This chapter focuses on hardware options you can consider from both IBM and third-party sources to increase the performance and capability of your PS/2 computer system. The chapter first examines several external hardware enhancements you can use with all members of the PS/2 family, such as printers, tape backup units, and mice. Next are descriptions of hardware enhancements that can be installed in the system unit of a PS/2. Because members of the PS/2 family can be categorized by the type of bus they use, the text also classifies adapter cards by their bus type. First you will learn about adapter cards designed for use in the PC bus of the Model 25 and Model 30 computers. Later on, adapter cards designed for use in PS/2 computers that incorporate Micro Channel architecture are covered.

Printers

Two of the primary methods for categorizing printers are by the interface they support and the method they use for type formation.

Types of Interface

Most printers are constructed with a *parallel interface.* This type of interface is designed to accept all of the bits that form a character at the same time. Hence the lines used to transmit data bits from the computer to the printer are routed in parallel in a cable connecting the two devices. The second type of printer interface is designed to accept data bits on one line, in a serial sequence. Thus, this type of interface is called a *serial interface.*

Each member of the PS/2 family contains one serial and one parallel port connector at the rear of the system unit. Normally the parallel port is cabled to a parallel-interface printer, whereas the serial port is commonly connected to an external modem. If you cable your serial port to a serial interface printer and also require the use of a modem or another serial-interfaced device, you must install an adapter card in your PS/2's system unit, which contains a second serial interface. Similarly, if you cable your parallel port to a parallel-interface printer but need to connect another parallel-interfaced device to your PS/2, you must install an adapter card in your system unit that contains a second parallel port.

Type Formation

The type formation of a printer can be categorized as fully formed character or dot matrix. A fully formed character printer uses a wheel that rotates to position a character between a hammer and the printer's ribbon. Then, the hammer strikes the wheel's character, resulting in a clean, sharp image of the character being printed.

In a dot-matrix printer, a printhead consisting of a matrix of pins is used to form characters. A microprocessor in the printer "fires" individual pins that hit the ribbon to form each character, normally resulting in a slight space between pin imprints. Because individual pins can be fired, a dot-matrix printer is capable of printing graphic images. As an example of this capability, consider the requirement to print a line graph. Using a dot-matrix printer, software can fire any pin in the printhead matrix to correspond to a specific point to be plotted. Thus, a printhead consisting of a matrix of 9 wires could fire any one of 9 pins vertically to best correspond to a specific point. In comparison, a fully formed character printer in which software uses the decimal point for plotting is restricted to plotting one vertical location for each character position. Thus, a fully formed character printer is not normally used to print graphic images.

Dot-matrix printers are commonly manufactured with 9-, 18-, and 24-wire printheads. Most modern dot-matrix printers are designed to provide a Near Letter Quality (NLQ) mode of operation. In NLQ printing the printhead traverses the paper first in one direction, firing its pins normally; the microprocessor then moves the paper vertically by a fraction of an inch and adjusts the horizontal timing of the pins. The printhead then traverses the paper in the opposite direction to repeat the row of characters. By printing in this manner, the spaces between the dots formed by the pins in the printhead are either reduced or eliminated. As you might infer, a 24-wire printhead normally produces a better NLQ type than an 18- or 9-wire printhead, because its wires are finer and are spaced more closely. Due to the improvement in NLQ printing technology, dot-matrix printers have basically relegated formed-character printers to law firms and other organizations that demand the sharpest, cleanest characters possible on all their documents.

Another advantage of dot-matrix printers over fully formed printers is print speed. Most fully formed printers have a print rate under 30 character per second (cps). In comparison, the slowest dot-matrix printers usually have a print rate much faster than the fastest fully formed printer, with some dot-matrix printers having a print rate above 300 cps.

IBM Printers

IBM manufactures numerous printers that can be used with members of the PS/2 family. This discussion examines two of that firm's more popular printer lines—the ProPrinter and Quietwriter.

The ProPrinter Series

The IBM ProPrinter II illustrated in Figure 2.1 is a low-priced printer intended for low to medium printing volumes. This printer has a 9-wire printhead and can print acceptable quality characters in its NLQ print mode.

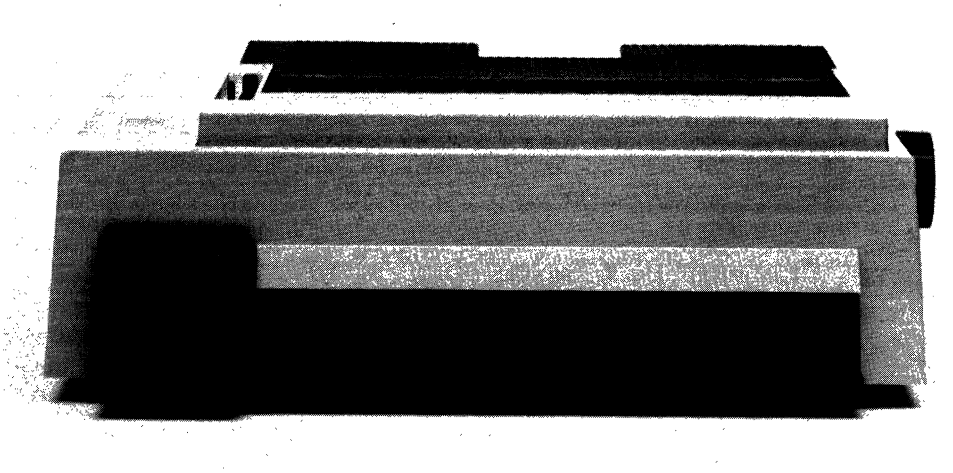


Figure 2.1
IBM ProPrinter II
(Photograph courtesy
of IBM Corporation)

The ProPrinter II can be set to print at one of four speeds. In its "Fastfont" mode, this printer can print as many as 240 characters per second (cps). In its DP mode, it can print as many as 200 cps, while in its Emphasized Print mode, the printing rate drops to 120 cps. Finally, when placed in its NLQ print mode of operation, the ProPrinter II achieves a maximum print rate of 40 cps.

Perhaps the most interesting feature of IBM ProPrinters is their ability to accept envelopes or single-sheet paper without the removal of previously inserted continuous paper. This is accomplished if you insert an envelope or a single sheet of paper end-up into the front slot in the printer. Then, by pressing the line feed button you can position the envelope or single sheet of paper to satisfy your printing requirement.

There are three other versions of the IBM ProPrinter series that warrant discussion. The ProPrinter XL is similar to the ProPrinter II, but has a wide carriage so you can print spreadsheets or charts as wide as 13.6 inches, in comparison to the 8½-inch paper width limitation of the ProPrinter II. The two other members of the ProPrinter series are the ProPrinter X24 and the ProPrinter XL24. Both of these printers use a 24-wire printhead, which results in a crisper, sharper NLQ print.

IBM Quietwriter

The IBM Quietwriter series of printers is designed for use in an office environment where a requirement exists for an executive's level of letter-quality production in a quiet environment.

The key to the silence and print quality of the Quietwriter is the use of resistive ribbon thermal transfer technology in each printer. Unlike the impact of pins hitting a ribbon in dot-matrix technology, the resistive ribbon thermal transfer technology uses an electric current applied to selective pins in a dot matrix to melt portions of the ribbon onto the paper. Because it uses electricity instead of the physical impact of pins on a ribbon, the Quietwriter's noise level is significantly lower than that of conventional dot-matrix printers, and its print image is improved as well.

Figure 2.2 illustrates the IBM Quietwriter III printer, the third IBM printer to use resistive ribbon thermal transfer technology. Similar to a ProPrinter, the Quietwriter III has an all points addressable (APA) graphics printing capability. The APA resolution of the Quietwriter is 240 × 240 dots per inch, in comparison to a maximum resolution of 180 vertical × 360 horizontal dots per inch on the ProPrinter X24 and XL24. For comparison purposes, it should be noted that the maximum resolution of standard laser printers is 300 × 300 dots per inch.

Tape Backup Units

The BACKUP and RESTORE commands included in the disk operating system (DOS) can be considered two of the most formidable obstacles to regularly performing fixed disk backup. To understand the problems associated with using floppy disks to back up data from a fixed disk, consider the user whose computer has a 20M byte fixed disk installed. Assuming that one 720K byte diskette drive is installed in the computer, a total of 28 diskettes would be required to back up 20M bytes of data. In comparison, a single or at most a few tapes can normally store the contents of a fixed disk.

Until the introduction of DOS 3.3, several deficiencies in the BACKUP command made life rather frustrating to users who lacked a tape backup unit. One potential frustration during a BACKUP procedure was the lack of enough previously formatted diskettes to hold the backed-up data. This situation forced users to another PC to format additional diskettes or else to terminate the BACKUP procedure and format the additional diskettes required. Unfortunately, under early versions of DOS, to invoke BACKUP a second time you had to restart the process from the beginning. Under DOS 3.3 and later versions of this operating system, unformatted diskettes can be formatted by the BACKUP command, thus removing one major source of potential frustration. While new versions of DOS eliminate the formatting problem, they do not reduce the large number of diskettes required to back up fixed disks with storage capacities of 20M bytes or more. Thus, the storage capacity of the fixed disk presents both an obstacle to good backup habits and a strong incentive to obtain a tape backup system.

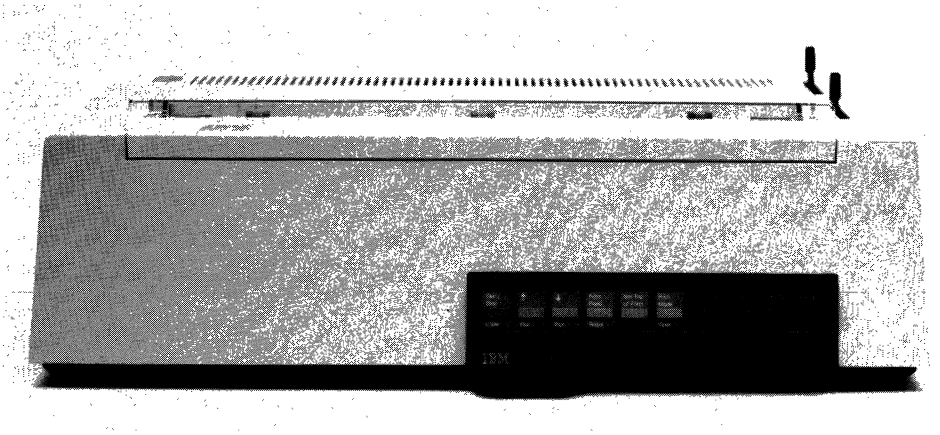


Figure 2.2
IBM Quietwriter III Printer (Photograph courtesy of IBM Corporation)

Types of Systems

Most tape backup units use industry standard ¼-inch tapes that are similar in appearance to audio cassettes. Other tape backup units are constructed to use 5¼-inch cartridges or ½-inch cartridge tapes. A typical 5¼-inch disk cartridge has the capacity to store 20M to 60M bytes of data, making the backup process of a fixed disk faster as well as requiring less storage space than floppy diskettes.

Figure 2.3 illustrates an external tape backup unit combined with a fixed disk installed in one device housing area. The system illustrated in Figure 2.3 is designed to be placed on top of the computer's system unit, next to the monitor, thereby requiring no additional desk space.

Method of Operation

In addition to internal and external systems, tape backup units can be classified according to their method of operation and interface. The two prevalent methods of operation are *start/stop* and *streamer*. In the start/stop method of operation, the tape moves only one record at a time during transfers from a fixed disk. Although a start/stop tape backup unit is preferable for applications such as selective backup because of its search-and-record capability, the extra precision for the unit's tape movement makes it the more expensive backup device.

The streamer tape backup unit is, as its name implies, designed for continuous recording. Streamer backup units are typically used to record the entire contents of a disk drive in a single session.

Typically, streamer tape backup units employ a microprocessor to control their servo mechanism. Although they have faster throughput and cost less than start/stop systems, they cannot perform selective retrievals.

Figure 2.3
Mountain Computer Series 7000 Combo System (Photograph courtesy of Mountain Computer)

Interface Considerations

The interface required to operate a tape backup unit varies considerably from one vendor to another. Many tape backup units require a separate controller and require a system expansion slot. Some tape backup units can be cabled to the serial port at the rear of the system unit, permitting one external tape backup unit to serve several personal computers. Although tape backup units connected to the serial port have a lower data-transfer rate than is obtainable through the use of a separate controller, the portability afforded by the use of this interface, and its ability to enable one unit to serve many computers, usually outweighs the extra minutes a backup procedure may require.

Software

The software provided by most tape backup unit manufacturers typically permits three types of backup: file-by-file, image, and by entire subdirectory. The *file-by-file* backup method can be used with either a start/stop or streamer unit. If a start/stop unit is used, the interrecord gaps waste tape. Normally, a file-by-file backup is used to selectively safeguard small amounts of data.

In an *image* backup, all data on a disk is recorded to tape. Many streamer units can operate at 90 inches per second, resulting in a backup of a full 20M byte disk in approximately 5 minutes.

Mouse

The pointing and selecting device first popularized by the Apple Macintosh has gained some acceptance by IBM personal computer users. This acceptance is due to the development of a variety of programs ranging in scope from desktop publishing to drawing and drafting that are enhanced by the use of a mouse.

IBM announced its PS/2 Mouse in April 1987, along with the original four members of the PS/2 family. The IBM PS/2 Mouse is a two-button device that is connected via a 9-foot cord to the pointing device connector located at the rear of the system unit of a PS/2 computer.

Operation

The IBM PS/2 Mouse, like most mice, incorporates a ball bearing in the mouse body; a small portion of the ball protrudes through a circular hole at the bottom of the housing. As the mouse is moved along a flat surface, the ball inside the housing moves. This ball movement is converted into a series of signals transmitted through the mouse cable to the computer.

To permit software to operate under the control of the movement of the mouse, you must install a *device driver*. IBM supplies a Microsoft Mouse compatible driver that works with such programs as Microsoft Windows and Microsoft Word to move a pointer on the screen to correspond with the movement of the mouse on the desktop.

Optical Mice

A second type of operation is obtained when you use an *optical mouse.* In place of a ball, the optical mouse uses a set of light-emitting diodes (LEDs) and optical receivers in conjunction with a reflective pad that contains grid lines. As the mouse is moved over the pad, the LEDs focus light on the pad and the receivers measure the reflectivity of the light. Because the reflectivity changes as the mouse passes over a grid line, its position on the pad can be determined. The advantage of an optical mouse is its lack of moving parts. This can be especially advantageous in an industrial environment, where dirt can cause a ball-based mouse to stick to its housing.

At the time this book was being written, only mechanical mice were available for use with PS/2 computers. Because the use of optical mice can prove to be highly advantageous in an industrial environment, you can expect this type of mouse to be marketed for use with PS/2 computers.

Internal Enhancements

This section examines hardware enhancement products you can install in the PS/2 system unit. Because members of the PS/2 family can be categorized by the bus structure they support, this discussion examines internal enhancements similarly. It begins with internal enhancements designed for use in the PC bus-based PS/2 Model 25 and Model 30 computer systems. Then the discussion moves to internal hardware enhancements designed for use with the Micro Channel PS/2 computers.

Model 25/30 Hardware Options

Table 2.1 lists 13 of the more common hardware options IBM markets for use with the PS/2 Model 25 and Model 30 systems. Because the system unit of the Model 25 and Model 30 computers can contain only two on-line storage devices, the selection of a second 3½-inch 720K byte diskette drive precludes the installation of a fixed disk and vice versa.

The IBM 3363 Optical Disk Drive contains a built-in laser that burns tiny spots on specially coated media contained in removable cartridges that are inserted into the drive. This technology provides you with the ability to *w*rite *o*nce on the media but to *r*ead it *m*any times; hence, another term used to describe this technology is *WORM.* The storage capacity of the disk cartridge used in the IBM 3363 Optical Disk Drive is 200M bytes, which is equivalent to over two hundred fifty 720K byte diskettes.

Figure 2.4 illustrates the standalone IBM 3363 Optical Disk Drive and a 5¼-inch optical disk cartridge. This disk drive is controlled by an adapter board inserted into a system expansion slot of a PS/2 computer. Two types of external optical disk drive systems are marketed by IBM; the difference between the two is the type of adapter card used in the PS/2 computer. The Model A01 Optical Disk Drive system includes an adapter designed for insertion into PC bus systems, whereas the Model A11 is designed for use in Micro Channel-based systems.

The IBM 2M byte Expanded Memory Adapter contains 2M bytes of expanded memory and a standard parallel printer port. This adapter can be used in PC bus computers

Table 2.1
Common PS/2 Model 25/30 Internal Options

Drives
3½-inch 720K byte diskette drive
3½-inch 20M byte fixed disk drive with adapter
3½-inch 20M byte fixed disk drive
3363 Model A1 Optical Disk Drive
Single-User Adapters
2M Byte Expanded Memory Adapter
3117 Scanner Adapter
PS/2 Display Adapter
Modems
PC 1200 bps Internal Modem
PC 2400 bps Internal Modem
Communications Adapters
Binary Synchronous Communications Adapter
SDLC Communications Adapter
Network Adapters
PC Network Adapter II
Token Ring Network Adapter

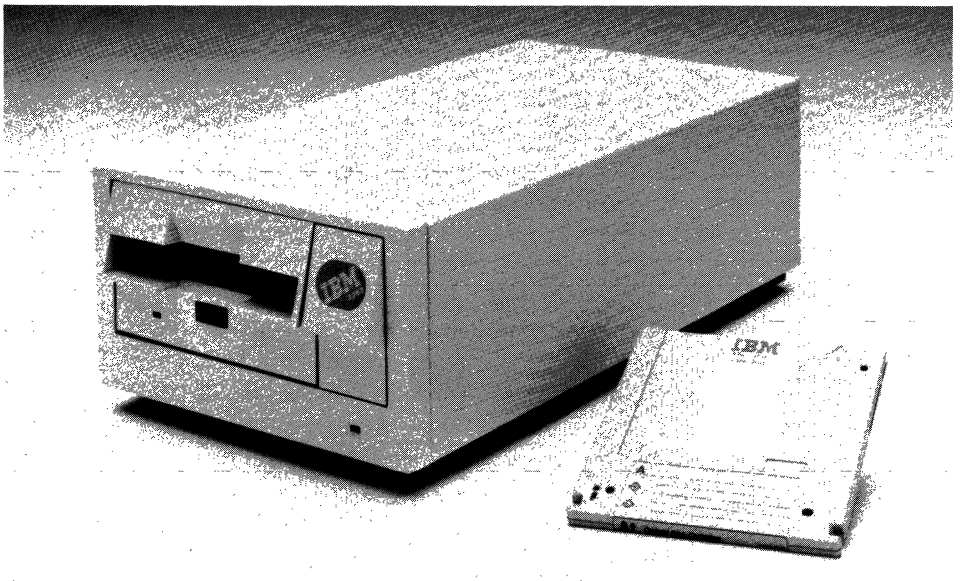


Figure 2.4
IBM 3363 Optical Disk Drive

to expand memory above the 640K bytes of maximum RAM installed on the system board of a Model 25 or Model 30 computer.

Using the IBM 3117 adapter you can attach an IBM 3117 Scanner to a PC-bus PS/2 computer. To obtain the same Video Graphics Array (VGA) capability built into Micro Channel based PS/2s and the PS/2 Model 30 286, you can install the IBM PS/2 Display Adapter in a PC expansion slot of a PS/2 Model 30.

The remaining six internal options listed in Table 2.1—including two internal modems—are all constructed on adapter cards designed for insertion into PC type expansion slots. IBM also markets similar adapter cards for installation in Micro Channel-based PS/2 computers.

The PC 1200 and PC 2400 internal modems provide asynchronous transmissions at data rates up to 1200 bits per second (bps) and 2400 bps, respectively. The remaining four adapters listed in Table 2.1 provide the ability for your computer to perform specialized communications functions. The IBM Binary Synchronous Communications Adapter and the SDLC Communications Adapter permit your computer to communicate with an IBM mainframe computer. The Binary Synchronous Communications Adapter supports IBM's older Binary Synchronous Communications protocol, whereas the SDLC Communications Adapter supports the more modern Synchronous Data Link Control (SDLC) communications protocol. The last two adapters listed in Table 2.1 enable PS/2 computers with PC expansion slots to connect to two different types of local area networks—the IBM PC Network and the IBM Token Ring network. For further information concerning PS/2 communications products, refer to Chapter 15.

Third-Party Products

The PS/2 Model 25 and Model 30 computers use the original PC bus architecture incorporated in the IBM PC and PC XT. Thus, literally hundreds of PC-compatible third-party products were also compatible with the Model 25 and Model 30 when these were introduced. Two of the more popular third-party products that users of PC-bus-based PS/2 computers may wish to consider are fixed-disk cards and VGA adapter cards.

Fixed Disk Card In late 1985, Plus Development Corporation introduced a low-powered 10M byte hard disk combined with controller circuitry, all mounted on an expansion card. Since then, approximately 20 vendors have introduced fixed-disk cards of varying capacities, which have become increasingly popular for several reasons. First, these disk cards can be installed in a system expansion slot without requiring the removal of a floppy diskette drive or another device previously installed in a device housing area. Thus, the use of a fixed disk card provides you with the ability to install two diskette drives and one fixed disk in the system unit of your PS/2. Second, the low power requirements of many disk cards enables them to be installed in most computers without replacing the power supply.

Figure 2.5 illustrates the Rodime R-Card 35. Note the edge connector on the card, which you insert into a system expansion slot just as you would any other type of adapter card. In addition to having storage capacity of 35M bytes, this plug-in fixed disk has an average seek time of 28 milliseconds, which is approximately one-third that of the full-height fixed disk marketed by IBM for use in the PS/2 Model 50. Because of its fast seek time, the Rodime disk card is well suited for use in a computer that functions as a server on a local area network.

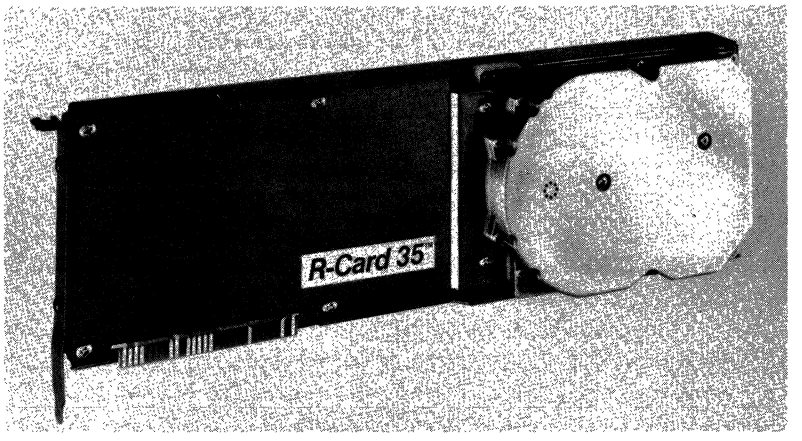


Figure 2.5
Rodime R-Card 35

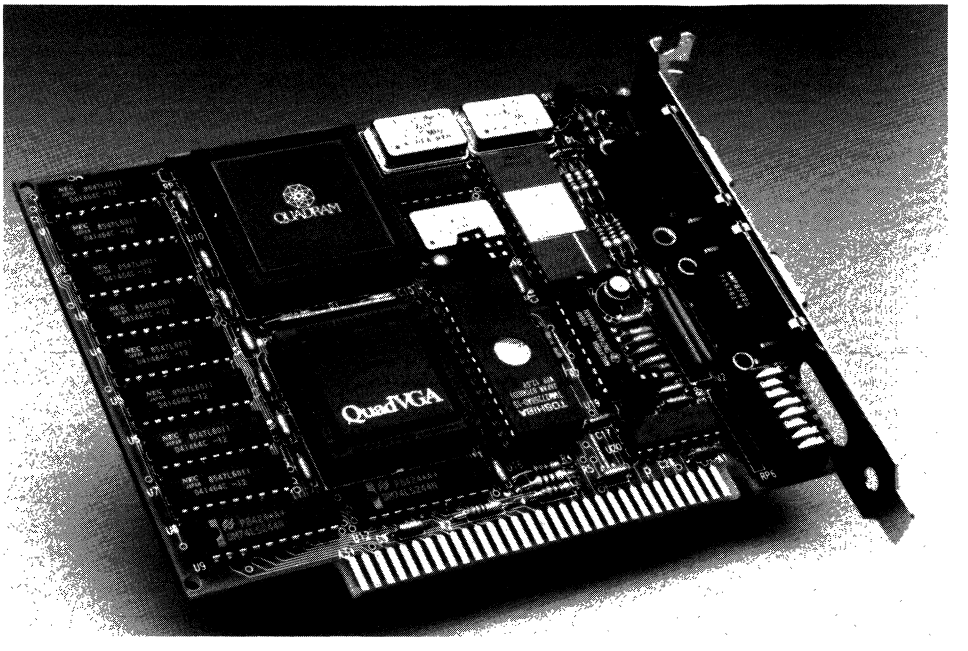


Figure 2.6
QuadVGA Graphics Adapter

VGA Adapter In addition to supporting IBM video display modes, some vendors also support the Hercules graphics display mode. Figure 2.6 illustrates the QuadVGA Graphics Adapter marketed by Quadram Corporation. The QuadVGA is based on the use of Chips & Technologies 441/442 chip set. This adapter supports all 17 VGA display standards, including what's known as Mode 13 (320 × 200 × 256 colors)—the only mode in which 256 colors can be displayed at once. The board runs all VGA modes at a vertical refresh rate of 70 Hz and also supports the popular Hercules graphics display mode.

One of the more interesting features of the QuadVGA is the inclusion of both a 9-pin digital and a 15-pin analog connector on the adapter. This software enables the QuadVGA to support a wide variety of monitors, such as IBM PS/2 Models 8503, 8512, 8513, and 8514, Princeton Graphic Systems' Ultrasync, NEC's MultiSync, Sony's MultiScan, and other variable scan monitors, as well as enhanced-display, color, and monochrome monitors.

Multifunction Boards Due to the limited number of expansion slots included in the PC bus-based members of the PS/2 family, it is quite easy to run out of expansion cards. To alleviate this situation, many third-party vendors have introduced multifunction boards that contain several functions on one card and require the use of a single expansion slot. Some multifunction boards combine a mixture of serial and parallel ports, whereas other boards combine video display and serial and/or parallel ports or memory and serial and/or parallel ports.

Micro Channel Hardware Options

Table 2.2 lists 20 of the more common IBM hardware options designed for use in PS/2 computers that have a Micro Channel bus architecture.

Table 2.2
Common PS/2 Micro Channel Computer Internal Options

Optional Equipment Varieties Available
3½-inch 1.44M byte Diskette Drive
PS/2 5¼-inch External Disk Drive
PS/2 Fixed Disk Drive
30M byte, 44M byte, 60M byte, 70M byte, 115M byte, 314M byte
3363 Model A11 Optical Disk Drive
Internal Optical Disk Drive
3117 Scanner Adapter /A
PS/2 80287 Math Coprocessor
PS/2 80387 Math Coprocessor
PS/2 80286 Expanded Memory Adapter /A
PS/2 80286 Memory Expansion Option
PS/2 80286 Memory Expansion Kit
PS/2 80386 Memory Expansion Option
PS/2 80386 System Board Memory Expansion Kit
PS/2 80386 Memory Expansion Kit
PS/2 300/1200 Internal Modem /A
3270 Connection Adapter
Token Ring Network Adapter /A
PC Network Broadband Adapter II /A
Dual Asynchronous Adapter /A
Multiprotocol Adapter /A

The 3½-inch 1.44M byte diskette drive provides 1.44M bytes of formatted storage capacity when high-capacity diskettes are used in this drive. In the drive's low-density recording mode, it is compatible with the 720K byte 3½-inch drive used in the Model 25 and Model 30 computers.

To provide PS/2 users with the ability to transfer data and programs between 5¼-inch media used with the older PC series of computers and the 3½-inch media used by members of the PS/2 family, IBM markets an optional external 5¼-inch diskette drive. This diskette drive must be installed as drive B of your computer. Then it operates in the same manner as a normal system B drive. Thus, to copy the contents of a 5¼-inch diskette to a 3½-inch diskette in drive A, you would enter the DOS command

```
COPY B:*.* A:
```

Refer to Chapter 4 for additional information concerning the 5¼-inch external disk drive, as well as information covering other methods of transferring information from a PC computer system to a PS/2 computer system.

As indicated in Table 2.2, IBM markets a variety of fixed disks that can be installed in the system unit of a PS/2 that uses the Micro Channel bus architecture. One fixed disk, the 60M byte drive, is designed as a replacement for the original fixed disk drive included with the PS/2 Models 50-021 and 50-031. Some fixed disks, such as the 44M byte and 70M byte drives, are designed to work with specific controllers either built into the system board or constructed on an adapter card that was installed in a Micro Channel system expansion slot as standard equipment. Thus, any user who desires to add more fixed disk storage capacity to a PS/2 computer should verify that the fixed-disk drive is supported by the fixed-disk controller in the computer.

Two types of optical disk drives are marketed for use with Micro Channel based PS/2 computers—internal and external units. The external unit is similar to that used with PC bus-based PS/2s and requires a controller constructed for use in a Micro Channel system expansion slot. The internal version of the standalone IBM Optical Disk Drive is only available for use in the PS/2 Models 60 and 80 computer systems.

The IBM 3117 Scanner Adapter /A is designed specifically to be installed in a Micro Channel system expansion slot. Similar to the 3117 adapter for PC bus computers, the 3117 Scanner Adapter /A provides you with the ability to attach an IBM 3117 Scanner to your Micro Channel PS/2 computer.

The 80287 math coprocessor is designed for insertion into the coprocessor socket of the system board of PS/2 computers that use the Intel 80286 microprocessor. The 80287 performs floating-point, extended-integer, and binary-coded decimal (BCD) operations. When installed, the coprocessor works in parallel with the 80286 microprocessor and enables your computer system to perform high-speed arithmetic, logarithmic, and trigonometric operations. The 80387 math coprocessor performs the same functions as the 80287, but works only with the Intel 80386 microprocessor.

The six memory products listed in Table 2.2 can be classified by their use in 80286 or 80386 computer systems. The 80286 Expanded Memory Adapter /A adds 2M bytes of usable memory to the system-board memory of PS/2 computers using the 80286 processor, such as the Models 50 and 60. The 80286 Memory Expansion Option is an adapter card designed for use with PS/2 computers that use the Intel 80286 microprocessor. This adapter card contains 512K bytes of RAM and can be expanded in

512K byte increments through the use of the 80286 Memory Expansion Kit, up to a total of 2M bytes of RAM. The 80386 Memory Expansion Option is a single full-length card designed for use in the PS/2 Model 70 and Model 80 computers. This card contains 2M bytes of 80-nanosecond RAM memory and can be expanded to hold a total of 6M bytes of RAM. This expansion is accomplished using the 80386 Memory Expansion Kit, which contains 2M bytes of RAM. The 80386 system board memory expansion kit, as its name implies, is used to expand RAM memory on the system board. This kit provides 1M bytes of 80-nanosecond RAM for the system board of the PS/2 Model 8580-041 computer, bringing total RAM on the system board to 2M bytes.

IBM's 80286 Expanded Memory Adapter /A can hold up to 8M bytes of 16-bit RAM memory. This board can be populated by the use of several types of memory module kits of varying capacity. Currently available memory kits for the 80286 Expanded Memory Adapter /A contain 0.5, 1M, and 2M bytes of 120-nanosecond RAM. Using these memory kits and the Expanded Memory Adapter /A, you can add a total of 8M bytes of RAM into one Micro Channel expansion slot of a PS/2 Model 50, Model 50Z, or Model 60. Up to two adapters can be installed in each computer, for a total memory of up to 16M bytes.

In a Micro Channel PS/2, the next four adapter cards listed at the bottom of Table 2.2 perform communications functions similar to those performed by the PC bus adapter cards described earlier in this chapter. Only the last two adapter cards listed in Table 2.2 perform communications functions unique to Micro Channel bus PS/2 computers.

The PS/2 Dual Asynchronous Adapter /A provides two independent serial communications ports on one card. Each port supports asynchronous communications at data rates from 50 to 19,200 bps and can be used to control such devices as an external modem, serial printer, serial plotter, or serial mouse.

The PS/2 Multi-Protocol Adapter /A provides the capability to communicate with a variety of devices that support different communications protocols, one device at a time. Instead of having to install separate adapter cards, you can use the Multi-Protocol Adapter /A card to transmit data in asynchronous, bisynchronous, and High-Level Data Link Control (HDLC) protocols.

Third-Party Hardware

Each PS/2 computer comes with one serial and one parallel port. Thus, to attach multiple serial or multiple parallel devices to your computer, you must install an appropriate adapter card that contains circuitry that provides an additional port or ports.

Although IBM markets an adapter card with dual ports, both ports on the card are asynchronous serial ports. To conserve system expansion slots, users requiring an additional serial and parallel port, as well as an expansion of RAM memory, might consider the use of the Quadram Quadboard PS/Q Multifunction Board. This board is illustrated in Figure 2.7. The adapter board is Micro Channel-bus compatible and provides the ability to add one parallel and one serial port as well as up to 4M bytes of RAM through one system expansion slot. Similar to IBM memory adapter cards, you can populate the Quadboard PS/Q Multifunction board using 0.5M byte or 1M byte memory kits. Unlike IBM memory adapter cards whose use is restricted to extended memory, the Quadboard PS/Q's memory is switch-selectable between extended memory and LIM specification expanded memory. This means that the memory on the board can be used

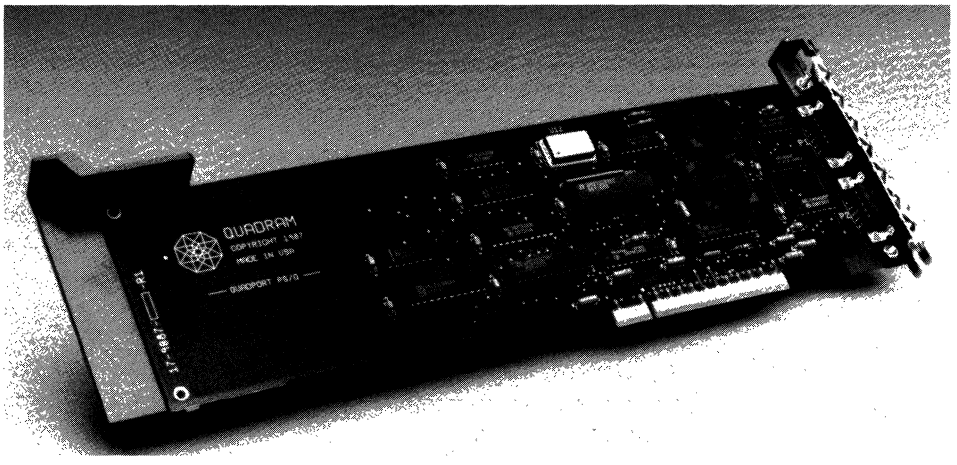


Figure 2.7
Quadboard PS/Q
(Photograph courtesy
of Quadram Corp.)

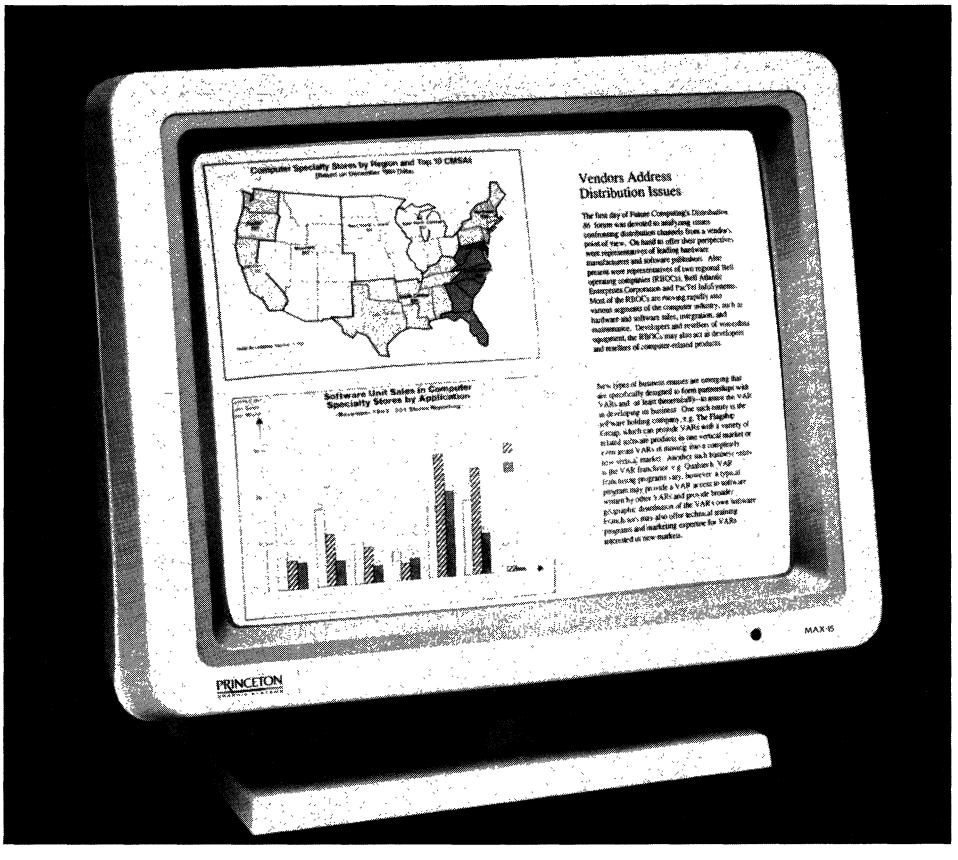


Figure 2.8
The Princeton
Graphics MAX-15
Monitor (Photograph
courtesy of Princeton
Graphics)

under OS/2 or you can use software that supports LIM bank switching (described in Chapter 1) under DOS to access memory above the DOS 640K byte barrier.

Another interesting third-party product designed for use with the IBM PS/2 family is the Princeton Graphic Systems MAX-15 autoasynchronous monochrome monitor. This monitor, which is illustrated in Figure 2.8, has one of the widest autoasynchronous horizontal and vertical scan ranges available, which makes it compatible with the IBM PS/2 family, the PC/XT/AT computers, and the Apple Macintosh II. The resolution of the MAX-15 is 1024 by 670 pixels, which provides an outstanding display resolution that can be used to display large "landscape" images when used in conjunction with appropriate software.

3 / Operating System Overview

This chapter focuses on each operating system (OS) you can use with members of the PS/2 family of personal computers. First the chapter explores the functions that an operating system performs. Next it examines the different operating systems from which PS/2 users can choose. This examination includes the characteristics and functions of each operating system, system hardware requirements, and the advantages and disadvantages associated with each OS.

OS Functions

An operating system can be viewed as a collection of programs that enable a computer to process commands and information entered by users, to supervise its own operations, and to execute such applications as spreadsheets and word processing programs. Performing functions similar to those of the central nervous system of the human body, an operating system controls all elements of the computer, including the interaction of software and hardware.

Each operating system consists of a nucleus of two elements—control programs and file-management programs. The control programs are responsible for the orderly and efficient flow of jobs through the computer, including controlling the scheduling and execution of processing programs.

The processing programs consist of language processors such as compilers and interpreters that compile or interpret source programs, as well as service programs that establish linkage between programs and application programs previously written using a language processor. Processing programs are invoked as required by control programs, which interpret commands entered by the user and arguments to be passed to the processing program.

The second key element of an operating system is a file-management system. Programs to manage files are used by the operating system to control the organization of data on different types of storage media, as well as to control recording and retrieval of data. Among the functions performed by the file-management system are formatting of storage media and creation and updating of tables that define the locations of files on the media.

Classes of Operating Systems

Some operating systems are designed for a single user who performs one task at a time. This type of OS is more formally known as a single-user, single-task operating

system. Currently, the single-user, single-task operating system is the primary operating system used with personal computers.

Single-User, Single-Task DOS

IBM's disk operating system (DOS) is a single-user, single-task operating system originally developed for the PC series of personal computers. Although DOS was introduced in 1981, it has been revised several times and for many PS/2 users will provide sufficient functionality and capability so they need not consider using the more recently developed OS/2 operating system.

Multiuser, Multitasking Operating System

Another category of operating systems is designed to control operations by multiple users. Such systems control multiple tasks concurrently. However, at any given moment only one task is actually being executed, because most computers only contain one arithmetic and logic unit—the processing unit of a computer. Known more formally as a multiuser, multitasking operating system, this class of OS assigns a slice of processing time to each task. If the task is not completed at the expiration of the time slice, the OS records parameters that define the state of the task and moves on to process the next task. As it moves on to process the next task, the OS may swap a task currently in memory to disk to obtain additional memory for processing, retrieve all or a portion of a previously saved task from disk, and update tables that define available storage and the status of tasks. This type of operating system is more complex than a single-user, single-task OS and was originally developed for use on mainframe computers to permit a large number of users to share the resources of hardware that could cost many millions of dollars.

Single-User, Multitasking OS/2

In between single-user, single-task and multiuser, multitasking operating systems is an operating system that combines a portion of the simplicity of the first OS with the capability of executing multiple tasks offered by the second type of system. Known as a single-user, multitasking operating system, this OS permits a computer user to execute one or more tasks concurrently. This type of OS enables a personal computer user, as an example, to write a letter using a word processing program while a previously constructed database program is sorting a file. To the user it appears that two simultaneous operations are being performed—word processing and database sorting. In reality, the operating system is switching between the two programs so rapidly that both appear to be executing simultaneously. IBM's Operating System/2 (OS/2), which was introduced with the PS/2 series, is a single-user, multitasking operating system.

Evolution of DOS

The operating system introduced with the PS/2 family was DOS 3.3. This operating system evolved from DOS 1.0, which was introduced with the original IBM PC in 1981. DOS 1.0 supported only single-sided 5¼-inch floppy disk drives and was rapidly superseded by Version 1.1, which added support for double-sided diskette drives.

When the IBM PC XT was introduced in 1983, a new version of DOS (Version 2.0) became available for use on both the PC and PC XT. Under DOS 2.0 and other minor releases of that operating system—including DOS 2.1—support was extended to hard disks. The 10M byte storage capacity of the fixed disk used by the PC XT was approximately 27 times the storage capacity of a 360K byte double-sided diskette used with that computer. This increase in storage capacity required the use of a hierarchical file structure for effective data management on the higher-capacity storage medium. Thus, a key difference between DOS 2.0 and previous releases of the operating system was the support of hierarchical file structures.

In 1984, IBM announced the PC AT, together with DOS Version 3.0. This version of the operating system added support for the PC AT's high-capacity 1.2M byte diskette drive, the capability to change diskette and disk volume labels used as storage medium identifiers, support for certain networking applications, and several additional features that are discussed later in this book. Releases 3.1 and 3.2 followed shortly thereafter, adding additional support for networking applications and, under DOS 3.2, support for 3½-inch diskettes, which had first been introduced in the laptop IBM PC Convertible.

The version of the operating system introduced concurrently with the announcement of the IBM PS/2 family of personal computers in April 1987 was DOS Version 3.3. This version of DOS differed primarily from Version 3.2 in the addition of support for different media introduced for the PS/2 family of personal computers, including the high-capacity 1.44M byte 3½-inch diskette drives that are standard on many models in that series.

In mid-1988, IBM introduced DOS 4.0, which included several significant enhancements over Version 3.3. These enhancements included a full-screen utility that simplifies the installation of the OS or its use as a replacement for a previously installed version of DOS, the addition of a DOS Shell, and the support of fixed disk partitions in excess of 32M bytes. Perhaps the most significant enhancement to DOS is its Shell. It offers a menu-oriented access to commonly used DOS commands and supports the use of a mouse. Users more comfortable with the DOS command language can toggle between using the DOS Shell and its Command Prompt mode, which accepts directly entered DOS commands.

Labeling Nomenclature

The nomenclature used by IBM in labeling versions of the OS takes the form *X.YZ*. Here the numeral in the *X* position denotes the major version under which to categorize that release of the operating system. The numeral in the *Y* position indicates a release containing substantial revisions to the major version, whereas the numeral for *Z* indicates a release with minor revisions to the major version.

Memory Use

DOS was originally developed to control the operation of computer systems that used the Intel 8088 microprocessor. The Intel 8088 has a 16-bit internal data path and for each memory operation generates a 20-bit address that can access 2^{20} (1,048,576) memory locations. However, the microprocessor has only eight pins for data, so it has to perform two memory operations to fetch or store a 16-bit data word, one byte at a time.

The Intel 8088 microprocessor computes every memory address in two parts: the *segment* and the *offset*. The segment is a contiguous area of memory, and the offset is the number of bytes from the start of the segment. To calculate a physical address, the 8088 multiplies the value of a segment by 16 and then adds the logical (offset) address, the result being a 20-bit address. This technique of using a segment and offset results from the 8088 address bus's limit of 16 lines, which precludes the direct addressing of more than 2^{16} or 64K bytes of memory.

Although the Intel 8088 can address up to 1024K bytes of memory, only 640K bytes of memory can be effectively used by DOS. This limitation results from the DOS requirement of a contiguous memory area, of which 640K bytes are always available based on the assigned usage of different areas of memory.

Figure 3.1 illustrates the System Memory Map for the PC and PS/2 family of computers, which explains why DOS only uses 640K bytes of memory. Note in the System Memory Map that beyond 640K bytes, memory is reserved for different types of video adapters, fixed disk control, power-on self-testing (POST), and other system functions.

When the IBM PC AT was introduced, a true 16-bit microprocessor was used that has 16-bit wide internal and external data buses providing the processing capability of the computer. This microprocessor, the Intel 80286, enables data to be manipulated in 16-bit units as well as sent to and received from memory in 16-bit units.

The 80286 bus has 16 data lines instead of the 8 lines used in the 8088-based PC and PC XT computers. The 80286 address bus consists of 24 lines, permitting the direct addressing of 2^{24} or 16M bytes of memory.

To be compatible with software developed for the IBM PC and PC XT, DOS was not revised to take advantage of the greater addressing capability of the Intel 80286 mi-

Figure 3.1
PC and PS/2 System Memory Map

Start Address Decimal	Start Address Hex	Function	
0	00000	Up to 640kb Read/Write Memory	
.	.		
640K	A0000	Enhanced Graphics	VIDEO
.	.		
704K	B0000	Monochrome	
720K	B4000		
736K	B8000	Color/Graphics	
752K	BC000		
768K	C0000	Professional Graphics	
784K	C4000		
800K	C8000	Fixed Disk Control	
816K	CC000	192K ROM Expansion and Control	
.	.		
.	.		
944K	EC000		
960K	F0000	Reserved (Cartridge ROM Area)	
976K	F4000	Base System ROM contains BIOS, Cassette BASIC and POST	
.	.		
1008K	FC000		

croprocessor. In addition, although the 80286 microprocessor was designed to perform multitasking and contained a design that facilitated memory management, DOS did not take advantage of these features included in the microprocessor's "protected" mode of operation. Instead, DOS controls the 80286 microprocessor in its "real" mode of operation, in which the chip essentially operates as an Intel 8088. This method of compatibility was extended to the PS/2 family of 80286 and 80386 computers. Computers with either microprocessor can execute DOS when they emulate the operation of an 8088 chip by running in their real mode.

Compatibility and Command Interface

One of the keys to the rapid acceptance of DOS is the downward compatibility of each release. That is, Version 4.0 could be used to operate programs developed under Version 3.X, and so on.

The command interface of DOS through Version 3.3 is text based as opposed to a graphics interface. Under a text-based interface, the personal computer user must enter instructions to the operating system in terms of commands formed by the use of alphanumeric characters.

Figure 3.2 illustrates the use of a text-based command interface. In this example, the command CD\WINDOWS changes the current directory on the fixed disk to the subdirectory named WINDOWS. The next command, DIR/W, results in a wide directory listing of the WINDOWS subdirectory. To use this command to obtain a wide directory listing requires you to remember to enter the /W switch after the command. Otherwise, DOS lists each entry in the WINDOWS subdirectory on an individual line, and the list rapidly scrolls off the screen unless you also incorporate the MORE filter in your command line.

Although the majority of DOS commands can be mastered without difficulty, casual users may prefer a graphics interface that provides the ability to highlight an operation and invoke it simply by pressing the Enter key. With the introduction of DOS 4.0, users obtained a graphics-based command interface as well as the ability to toggle between the command prompt mode and the graphics mode.

Using a graphics interface, you move a pointer onto an icon and select the icon by either pressing a key combination or clicking the button on a mouse or other pointing

Figure 3.2 Text-Based Command Interface

```
C>cd\windows

C>dir/w

 Volume in drive C is DATACOM
 Directory of  C:\WINDOWS

                 ..                  PIF                 WIN       COM     WIN200    BIN
WIN200    OVL    SPOOLER   EXE       WINOLDAP  MOD       EPSONLQ2  DRV     CLIPBRD   EXE
READMEEP  TXT    MODERN    FON       SCRIPT    FON       ROMAN     FON     TMSRB     FON
COURB     FON    HELVB     FON       TMSRC     FON       TMSRA     FON     COURC     FON
COURA     FON    HELVC     FON       HELVA     FON       CALC      EXE     CALENDAR  EXE
CARDFILE  EXE    CLOCK     EXE       CONTROL   EXE       CVTPAINT  EXE     NOTEPAD   EXE
PAINT     EXE    REVERSI   EXE       TERMINAL  EXE       DOTHIS    TXT     ABC       TXT
WRITE     EXE    PIFEDIT   EXE       README    TXT       PRACTICE  WRI     WINOLDAP  GRB
MSDOS     EXE    WIN       INI       A&L
       43 File(s)  24887296 bytes free

C>
```

device. Figure 3.3 illustrates the WINDOWS subdirectory viewed through the use of Microsoft Windows. In this example, the white arrow represents the mouse pointer, used to select a file called A&L, highlighting the filename in reverse video.

Figure 3.3 also shows that the icon for drive C is highlighted, informing you that this operating system is working with that drive. To the right of the icon representing drive C is the text C: \WINDOWS, which informs you that you are in the subdirectory WINDOWS on the fixed disk. By double clicking on a selected program file, you execute the file. In a text-based command interface, by contrast, you would have to enter the appropriate command from the keyboard to execute the desired program.

Figure 3.4 illustrates another key feature of a graphics-based command interface—drop down windows. In this example, the mouse pointer was moved to `Special`, causing the options available for selection to drop down in the form of a window. Note that under Microsoft Windows, you can either move the pointer to an appropriate row in the drop down window and double click to select the option, or you can enter the character underlined in the row to select the option. Thus, you could enter the character **C** to select the `Change Directory` option displayed in the Special window.

For comparison purposes, Figure 3.5 illustrates the DOS 4.0 File System when the DOS Shell is used to produce a graphics-based command system. Note that the DOS Shell not only displays icons for each disk drive, but also provides both a directory tree of your selected disk (left portion of Figure 3.5) and a listing of the files in the current directory (right portion of Figure 3.5). Under the DOS Shell you can shift to entering command prompts by pressing Shift+F9. By pressing the F10 key, you can enter the letter F, O, A, or X for Exit, each of which displays a different window containing options for selection.

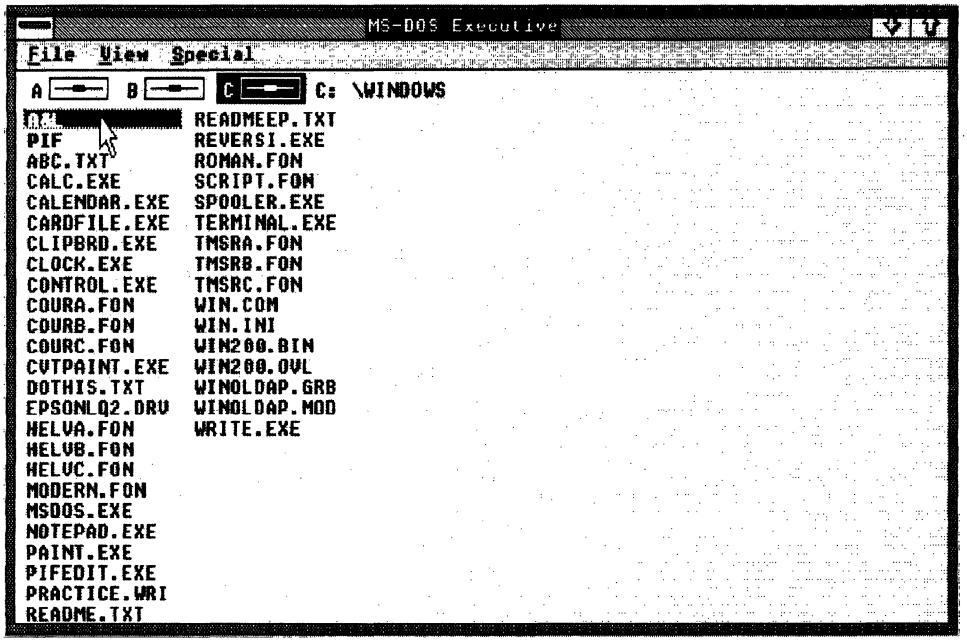


Figure 3.3
Using Microsoft Windows to Perform a File Selection

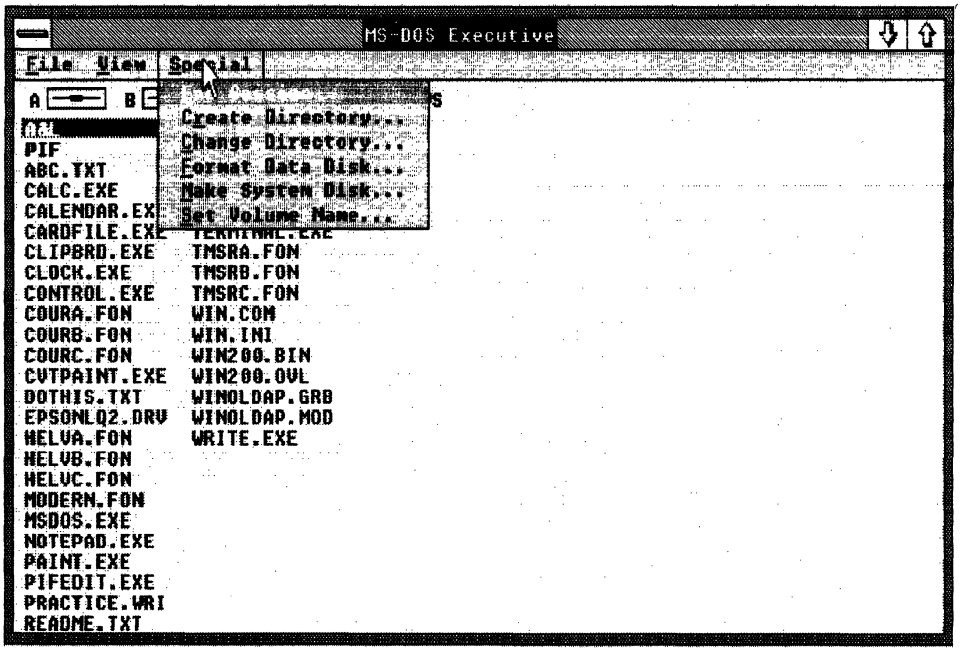


Figure 3.4
Microsoft Windows' Drop Down Windows

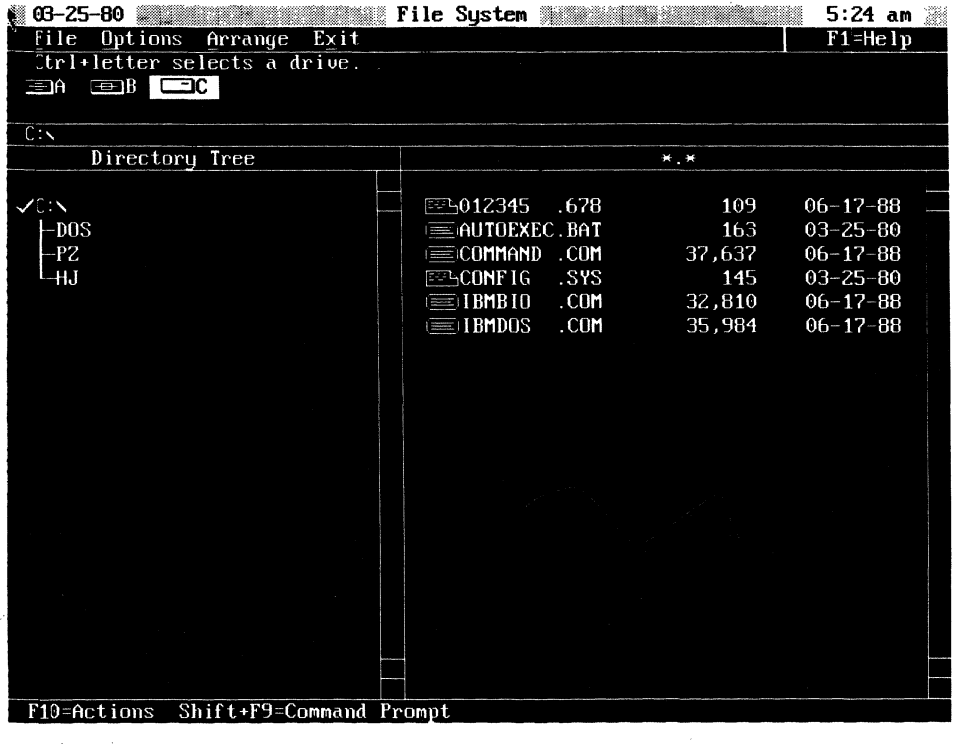


Figure 3.5
DOS 4.0 Shell File System

The DOS 4.0 Shell graphics interface supports the use of a mouse, just as Microsoft Windows does. It differs from Microsoft Windows by including context-sensitive help that can be displayed if you press the F1 key.

OS/2

IBM's OS/2 was developed as a growth or migration path for many DOS users, as well as a preferred operating system for certain categories of new PS/2 users. Two of the major differences between DOS and OS/2 are the latter's ability to perform multitasking and its ability to break the 640K byte DOS memory barrier. Unfortunately, this added capability is not without substantial cost, because OS/2 requires a substantial amount of RAM memory as well as hard disk space.

OS/2 Versions

So far, IBM has announced four versions of OS/2. IBM's OS/2 Standard Edition 1.0 can be viewed more as a development and trial tool, because its primary usage was by organizations that wanted to evaluate the level of performance of this operating system in comparison to DOS.

OS/2 Standard Edition 1.0 has a text-based command interface and was superseded by Version 1.1, which includes the OS/2 Presentation Manager, a graphics interface based on the Microsoft Corporation Windows Presentation Manager.

IBM's Extended Edition 1.0 includes all of the functionality of the Standard Edition 1.0, plus built-in database and communications management programs. This version of OS/2, in effect, bundles both database and communications into the operating system and will primarily appeal to organizations that can use the additional capability of these programs.

Like the Standard Edition 1.0, the Extended Edition 1.0 can be viewed as an interim operating system that was superseded by Extended Edition 1.1. This version of OS/2 incorporates the Presentation Manager graphic user interface and is designed for organizations that prefer to obtain both database and communications functions bundled with the operating system instead of obtaining such capability as separate entities.

All versions of OS/2 are based on the same kernel or core of instructions and features contained in the code used to develop the Standard Edition 1.0. The kernel contains the basic functions of OS/2 to include multitasking, protected mode operation for applications that require more than 640K bytes of memory and DOS compatibility. Figure 3.6 illustrates the functional relationship between the four versions of OS/2.

Each version of OS/2, in effect, is a dual-mode operating system. In its DOS mode of operation, you can execute only one application at a time, whereas in the OS/2 mode you can execute multitasking applications.

Regardless of the mode of operation, files created under the control of one operating system are fully compatible with the other operating system. This data file interchangeability provides an easy migration path from DOS to OS/2, as well as the ability to share files among computers operating under DOS and OS/2.

The text command interface of OS/2 is very similar to DOS. In addition, a majority of the commands supported by OS/2 are identical to DOS commands, whereas many

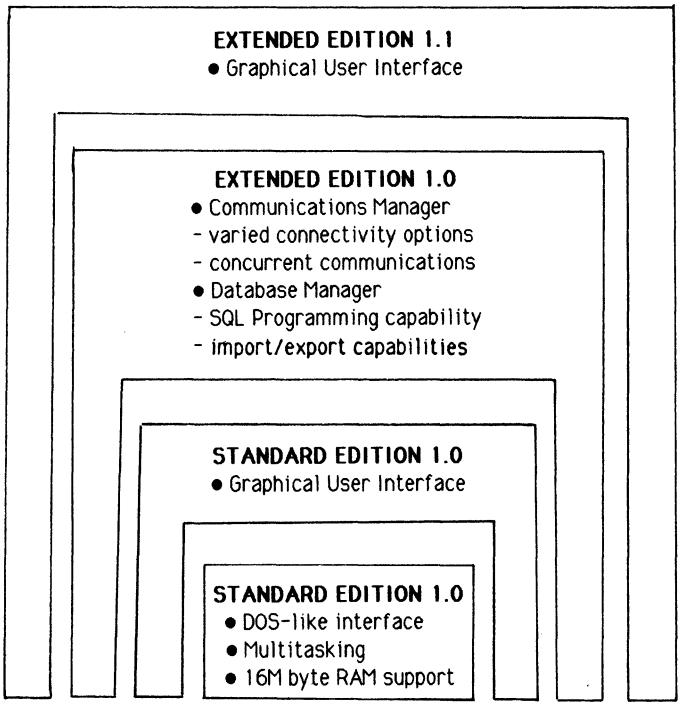


Figure 3.6
Comparison of OS/2 Editions' Functions

other OS/2 commands are very similar to DOS commands, usually varying slightly in form or by the parameters that are supported by the command. In fact, less than one quarter of the combined total of DOS and OS/2 commands are specific to one operating system.

Because OS/2 was designed to perform many core functions performed by DOS to include entering the time and date, copying files, and navigating hierarchical directory structures, users who understand DOS can easily migrate to the advanced features of OS/2. Due to this, the book examines the structure and operation of DOS commands in subsequent chapters prior to examining OS/2.

Although each version of OS/2 supports applications developed to execute under DOS, the real efficiency of the newer operating system becomes apparent when you execute programs specifically developed for OS/2. Unlike DOS, OS/2 application programs do not reference physical addresses. Instead, programs written to operate under OS/2 use virtual addresses that OS/2 converts just before the microprocessor converts the addresses to physical addresses. This method of memory management permits RAM to be used more effectively than under DOS. As an example, under DOS an application would reserve all free memory and use all or a portion of that memory as the program executed. Under OS/2, only the memory required at a particular time is assigned to an application. This permits the operating system to control memory more effectively, permitting two or more programs to operate concurrently using less RAM than if they were executed on separate personal computers under DOS.

Comparing Operating Systems

Table 3.1 compares seven major features between DOS and four versions of OS/2.

The key to the multitasking capability of OS/2 is its ability to operate the Intel 80286 and 80386 microprocessors in their protected mode of operation. Unfortunately, this precludes OS/2 from being used on the Intel 8086-based PS/2 Model 25 and Model 30, because that microprocessor is essentially a 16-bit bus 8088 and does not support protected mode operations. In addition, OS/2 cannot operate on the Intel 8088-based IBM PC and PC XT computer systems. Thus, to use OS/2, your computer must use an 80286 or 80386 microprocessor, such as the PS/2 Models 50, 60, 70, and 80 and the PS/2 Model 30 286.

The support of protected mode operation permits OS/2 to operate 80286 and 80386 microprocessors using a 24-bit address bus. This enables OS/2 to directly address 16M bytes, whereas DOS restricts memory support to 640K bytes.

Note in the table the significant differences in minimum recommended RAM and hard disk space between DOS and the various versions of OS/2. Although the minimum memory of PS/2 computers is 1M byte, most PS/2 users operating OS/2 probably require at least 1M byte of additional RAM for effective operations. This is because OS/2 takes up approximately 500K of RAM, leaving 0.5M bytes on a 1M byte system to take advantage of the multitasking capability of the operating system. Because two OS/2 application programs could each consume 0.5 to 1M of RAM, 2.5M bytes is

Table 3.1
Operating System Features and Price Comparison

		OS/2			
		Standard Edition		Extended Edition	
Feature	**DOS**	**1.0**	**1.1**	**1.0**	**1.1**
Microprocessor Support					
8086	yes	no	no	no	no
8088	yes	no	no	no	no
80286	yes	yes	yes	yes	yes
80386	yes	yes	yes	yes	yes
Multitasking	no	yes	yes	yes	yes
Minimum recommended RAM (bytes)	256K bytes	1.5M bytes	1.5M bytes	1.5M bytes	1.5M bytes
Hard disk space recommended	not required	2.5M bytes	2.5M bytes	2.5M bytes	2.5M bytes
Addressable memory (bytes)	640K bytes	16M bytes	16M bytes	16M bytes	16M bytes
Virtual memory support	no	yes	yes	yes	yes
Cost	$125	$325	—[1]	$795	—[2]

1. Licensees of Standard Edition 1.0 can obtain a copy of version 1.1 at no charge.
2. Licensees of Extended Edition 1.0 can obtain a copy of version 1.1 at no charge.

probably a more realistic memory capacity to use the multitasking capability of this operating system effectively.

When the additional hardware costs of OS/2 to include RAM and perhaps a larger fixed disk are added to the cost of the operating environment, significant differences between running OS/2 and DOS become apparent.

The actual cost difference between OS/2 and DOS actually depends on the version of OS/2 used, the amount of RAM installed to support multitasking, and (usually) the cost of obtaining a higher-capacity hard disk. In general, OS/2 users can expect to spend between $1500 and $3000 more than DOS users on hardware and software.

OS/2 Alternatives

OS/2 readers can consider a variety of alternatives, with the governing factor being your requirement for one or more OS/2 features. Table 3.2 lists software and hardware alternatives to OS/2 that can provide many of the features of this operating system.

If your primary concern is the ease of use provided by a graphics interface, consider investing in Microsoft's Windows, Windows/386, or Digital Research Corporation's GEM. Each of these programs is actually an operating environment that overlays DOS. Both Windows and GEM operate on any PC or PS/2 computer, although they work best with 80286- or 80386-based machines. Windows/386 was designed specifically for computers containing the Intel 80386 microprocessor and takes advantage of the "virtual" mode feature of that microprocessor. This feature lets the microprocessor function as if it were multiple 8086 chips, permitting each window to act as its own "virtual machine," resulting in both multitasking and protection for multiple applications.

Although GEM is a single-user, single-task operating environment, versions of Windows—as well as Quarterdeck's DESQview, Software Link's PC-MOS/386, and several other commercially available programs—can provide a multitasking environment.

Some operating environments, most notably DESQview, can run almost all applications previously developed for operation under DOS without modification. Other operating environments, such as Windows and GEM, are designed for use with specific programs written to take advantage of the common user interface developed for each environment. Due to this fact, you may have to replace some or all of your application

Table 3.2
OS/2 Alternatives

Software	
Function	*Program/OS to Consider*
Graphics interface	Microsoft Windows, Windows/386 Digital Research GEM
Multitasking capability	Quarterdeck's DESQview Software Link PC-MOS/386 Microsoft Windows, Windows/386
Hardware	
Function	*Hardware to Consider*
Expanded memory	LIM compatible memory
Faster processing	Accelerator boards

programs with programs developed to operate under a specific operating environment to maximize the operational functionality provided by the environment.

Two types of adapter cards can provide DOS users with some of the functionality provided by OS/2. First, you can obtain an extended memory capability for executing large programs or for performing multitasking operations without requiring OS/2 to operate an 80286 or 80386 microprocessor in its protected mode. This can be accomplished by the installation of expanded memory.

Expanded memory can be used with any microprocessor-based personal computer, because it accesses segments of memory above the 1M byte boundary. It uses a technique known as *paging* or *bank switching.* Bank switching is the process of electronically repositioning expanded memory into the microprocessor's address range. The expanded memory is divided into 16K byte blocks called *pages,* which are swapped into or out of an area of main storage called the *page frame.* The page frame effectively becomes a window that can look into various blocks of expanded memory, as previously illustrated in Figure 1.6. See Chapter 1 for additional information concerning expanded memory.

The second hardware item listed in Table 3.2 can be used as an alternative to OS/2's high performance as well as a mechanism to operate OS/2 on personal computers that do not use such protected mode microprocessors as the 80286 and 80386. Known as *accelerator boards,* these adapter cards include a microprocessor and RAM memory. To use the accelerator board, you first remove the microprocessor installed in your computer and then install the adapter card in an expansion slot. After you cable the adapter card to the socket formed by the removal of the computer's microprocessor, the chip on the card provides the processing power for your system.

Accelerator boards have been manufactured with 8086, 80286, and 80386 microprocessors. Although all accelerator boards provide faster processing than 8088-based PC and PC XT systems, only 80286- and 80386-based boards can be used to run OS/2. Such boards can be installed in PCs, PC XTs, and such PS/2 systems as the Model 25 and Model 30, which normally are incapable of supporting protected mode operation. Thus, you can also use an accelerator board to obtain the ability to use OS/2 on a computer system that was originally manufactured with a microprocessor capable of supporting only real mode operations.

4 / System Setup and File Transfer Tips

This chapter prepares you to set up the PS/2 computer for operation. Topics in the first portion of this chapter include setting up a desktop PS/2, the use of an AC power protector, the installation of options in the system unit of the computer, and how to effectively use the Reference Disk that is shipped with each PS/2. The second portion of this chapter focuses on the different disk media supported by personal computers. By examining the storage capacity of different types of diskettes, you will see how file transfer problems can result when you attempt to transfer programs or data from one computer to a computer that uses a different disk medium.

Then, the chapter explores six file transfer methods that can alleviate media incompatibilities. After you master this chapter's material, you should be able to transfer programs and data from IBM PC and compatible personal computers to PS/2 computers, and vice versa.

PS/2 Power Requirements

Normally a dual power receptacle is inadequate for your power requirements. This is because if you have several options that are independently powered, such as a printer, external modem, and plotter, a dual power receptacle will be insufficient to satisfy your power requirements. For this type of situation you may want to use a *power strip,* connecting the power strip to your household receptacle to obtain several additional outlets in one convenient location.

For most computer and peripheral configurations, a standard six-plug power strip is recommended. Some power strips include circuit breaker protection against overloads; however, they do *not* protect your equipment against high-voltage spikes that occur from lightning or power surges caused by machinery operating in close proximity to your computer. For added protection against possible electrical damage, be sure your power strip or other type of multiple outlet device includes an AC power protection capability.

AC Power Protector

Although devices that protect a personal computer from spikes, surges, noise, and brownouts are not necessary for the operation of the computer, they are one of the first accessories you should consider.

AC power protector products include surge suppressors, isolation transformers, and voltage regulators. Some devices perform just one of these functions; others may perform two or all three of the functions.

The *surge suppressor* rejects high-amplitude spikes and, depending on manufacturer, may reject radio frequency interference (RFI) noise.

An *isolation transformer* provides a higher degree of spike filtration than a surge suppressor and also rejects RFI noise.

The most sophisticated AC power protector is a *voltage regulator.* This device includes a transformer that maintains a constant output voltage over a wide range of input voltages. It works fast enough to eliminate sudden as well as gradual surges and sags. Also, the voltage regulator suppresses short spikes, as well as RFI that is carried by house or office wiring. The independent windings isolate the load equipment from the house or office wiring and provide a high degree of protection against electrical shock in the event of a catastrophic malfunction in the load equipment's own power supplies.

Figure 4.1 illustrates the SSB Design Pure Power Plus, which is a combined multiple outlet, surge suppressor, and isolation transformer. The SSB Design Pure Power Plus provides six 115 VAC electrical outlets at the rear of the device whose power activity is controlled by individual switches at the front of the unit. Although several manufacturers make similar devices, the current indicator at the front of the unit is probably unique and provides you with a visual indication of how much current your equipment is using. The voltage indicator tells you the condition of the line voltage, which can be

Figure 4.1
The SSB Design Pure Power Plus (Photograph courtesy of SSB Design)

especially valuable during the summer months, when electric utilities have been known to reduce voltages during periods of peak consumer demand.

In addition to protecting against spikes and surges, the SSB Design Pure Power Plus includes electromagnetic interference (EMI) and RFI filtration. Because noise in the form of EMI or RFI can result in memory errors, this feature may be valuable for personal computers located in an industrial area.

PS/2 Setup

When you install your system for the first time, install certain options in your system unit, or replace your computer's battery, you must perform certain operations so your computer can operate correctly. The key to correctly setting up your computer is the IBM Reference Diskette. This diskette contains programs written to acquaint you with the features of your PS/2 as well as programs you can use to establish password access to your computer, set or reset the date and time maintained by your computer, set the configuration of your computer, and perform other functions that will govern PS/2 operations.

The Reference Diskette

The Reference Diskette is a permanently write-protected diskette. This means that you cannot record information onto the diskette. Because the addition of certain optional equipment requires you to transfer configuration information from diskettes furnished with the equipment to the Reference Diskette, one of the first actions you should take after you load the Reference Diskette is to make a copy of that diskette. Before you actually work with the Reference Diskette, first review the system unit of your PS/2 and install any required hardware options in the computer.

System Unit

Figure 4.2 illustrates the rear of the system unit of a desktop PS/2. There are some slight differences in the back of the system among members of the PS/2 family. As an example, the expansion slots on the Model 30 are constructed horizontally, whereas they are vertical on the Model 50 and Model 70 computers.

Installing Options

Installation of many types of adapter boards into the system unit of a PS/2 computer will require you to update a backup copy of the Reference Diskette to make your system operate correctly. Due to this, you will next see how to install adapter boards. For information concerning the installation of other options, such as diskette and fixed disk drivers, refer to the quick reference guide provided by IBM with each PS/2 computer.

Prior to removing the cover from the system unit, you should disconnect the keyboard and monitor if they were previously connected; set them aside. Next, ensure that the system unit power switch is off and then disconnect the power cord and all cables from the rear of the system unit as an added safety precaution.

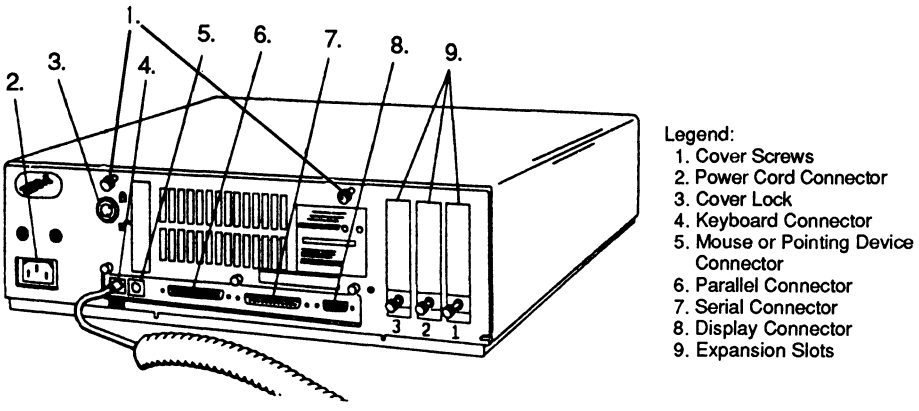


Figure 4.2
Rear of Desktop System Unit

Check that the cover lock is unlocked. If it isn't, use the tubular key that came with your computer to switch the cover lock to the unlocked position, facing downward. Next, loosen the two cover screws located at the rear of the system unit using a flat blade screwdriver or coin. Then you can remove the cover to the system unit by sliding it forward approximately two inches and lifting it.

Prior to installing an optional adapter, read the instructions furnished with it to determine whether it can be installed in any available expansion slot or if it requires a specific expansion slot. Before you install the adapter, remove the expansion slot cover. This can be accomplished by turning and removing the expansion slot cover screw that holds the cover in place, using a coin or flat blade screwdriver if the screw is too tight. After the expansion slot cover screw is unfastened, you can lift the expansion slot cover upward, removing it from the system unit.

Although IBM's quick reference guide says you can discard the expansion slot cover, it is a good idea to place it in a location where you can easily locate it at a later date. You may never need it, but if for some reason you remove a previously installed adapter, you will have a rectangular hole in the rear of your system unit that allows dust and dirt into the unit. Thus, keep the cover so you can close the expansion slot if you decide to remove a previously installed adapter.

Adapters manufactured for use in the PS/2 contain a built-in expansion slot cover. You install the adapter by first positioning it so the end with the expansion slot cover faces the rear of the system unit. Then press the adapter firmly into the expansion slot connector until the adapter clicks into place. Using the screw obtained when you removed the expansion slot cover you can fasten the adapter card to the system unit. At this time you should install the system unit cover, tighten the cover screws, and connect your monitor and keyboard to the system unit. Once this is accomplished, you are ready to use the IBM Reference Diskette to complete the setup of your computer.

Using the Reference Disk

Prior to turning on power to your system unit, insert the IBM Reference Diskette in drive A. Drive A is the leftmost diskette drive if your computer has two diskette drives. If your computer has only one diskette drive, that drive is drive A.

If everything is ready, when you turn the power switch on the system unit to the upright, On position, three things will happen. First, the keyboard status lights blink on and off. Next, a memory test occurs, and the amount of memory installed in your system unit followed by the letters OK should be displayed in the upper left corner of your screen. After the power-on self-test is completed, you should hear one short beep, indicating that your system unit has completed its self-test.

The time required to perform the self-test depends on the amount of memory installed in the system unit and can vary from 10 to 30 seconds. If you do not hear one short beep, your system unit may contain a defect, and you may wish to contact the source where you purchased the computer or try to run the Test the computer option from the Main Menu of the Reference Diskette. If error message number 165 is displayed and you hear two short beeps, do not be alarmed. This is the method by which the PS/2 computer notifies you that its configuration is not set correctly and that you must correct this situation before you can use the operating system.

Once you place the Reference Diskette in drive A and turn on power to your system unit and monitor, the IBM logo displays. Pressing the Enter key causes the Main Menu of the Reference Diskette to display, illustrated in Figure 4.3.

You can select an entry from the Reference Diskette Main Menu by pressing the Up Arrow or Down Arrow key to move a highlighting bar over the desired option. Then you can press the Enter key to invoke the selection.

Selecting option 1 invokes a program that displays elementary information about hardware, software, communications, and testing the computer. For those using a computer for the first time, this selection will provide you with basic information that more experienced persons will probably skip.

Option 2, Backup the Reference Diskette, helps you to remove a 165 error message and to set your computer's configuration after you install one or more adapter boards in the system unit. Because the Reference Diskette is write protected, you must copy its contents to a backup diskette so you can add a configuration file or files to the diskette. After you select option 2, the program will request you to insert a backup diskette in your disk drive. The program on the Reference Diskette will first format the backup diskette to prepare it for recording data and then transfer the contents of the Reference Diskette to the backup diskette, prompting you several times to insert

Figure 4.3
Reference Diskette Main Menu

```
Main Menu

 1. Learn about the computer
 2. Backup the Reference Diskette
 3. Set configuration
 4. Set features
 5. Copy an option diskette
 6. Move the computer
 7. Test the computer
```

and remove diskettes if you have a system with a single diskette drive. If you had a 165 error message, once this activity is completed, select option 5 from the Main Menu.

The `Copy an option diskette` selection enables you to copy configuration file information from option diskettes included with certain adapter cards to the previously created backup copy of the Reference Diskette. These files are required for you to successfully use the Main Menu's `Set configuration` option.

Selecting option 3 from the Reference Diskette Main Menu results in the display of the Set Configuration menu screen illustrated in Figure 4.4. As indicated in Figure 4.4, you can select five options from this menu.

Selecting option 1 from the Set Configuration menu provides a display of your current configuration. This display is similar to that illustrated in Figures 4.5 and 4.6, with the key difference between the data displayed in those figures and what is actually displayed on your computer screen resulting from whatever features are installed in your system unit. In addition, if you previously installed one or more adapter cards in the system

Figure 4.4
Reference Diskette Set Configuration Menu

```
Main Menu

   Set Configuration

   1. View configuration
   2. Change configuration
   3. Backup configuration
   4. Restore configuration
   5. Run automatic configuration

U
E  Press a number to select.
   Esc=Quit    F1=Help
```

Figure 4.5
First Screen of View Configuration

```
View Configuration

 Total System Memory
    Installed Memory ..................... 1024KB (1.0MB)
    Usable Memory ........................ 1024KB (1.0MB)

 Built In Features
    Installed Memory ..................... 1024KB (1.0MB)
    Diskette Drive A Type ................ 1.44MB 3.5"
    Diskette Drive B Type ................ Not Installed
    Math Coprocessor ..................... Not Installed
    Serial Port .......................... SERIAL_1
    Parallel Port ........................ PARALLEL_1

 Slot1 - The IBM 3270 Connection Version B
    Resources Used ....................... First
    Memory Assignments ................... CE00

 Slot2 - IBM Token-Ring Network Adapter/A

Esc=Quit
F1=Help                                  End    PageDown
```

Figure 4.6
Second Screen of View Configuration

```
View Configuration

    Serial Port ..................... SERIAL_1
    Parallel Port ................... PARALLEL_1

  Slot1 - The IBM 3270 Connection Version B
    Resources Used .................. First
    Memory Assignments .............. CE00

  Slot2 - IBM Token Ring Network Adapter/A
    Primary or Alternate adapter .... Primary
    ROM Address range ............... CC000 - CDFFF
    RAM Address range ............... D8000 - DBFFF
    Interrupt Level ................. Interrupt 2

  Slot3 - Empty

  Slot4 - IBM Fixed Disk Adapter
    Type of drive ....................  30
    Arbitration Level ............... Level_3

Home   PageUp                    |
F1=Help                           End
```

expansion slots of your computer, information concerning the use of the slot will not be displayed. This information must be stored in what is known as complementary metal oxide semiconductor (CMOS) memory, which is battery-powered memory inside your system unit and which is critical for the correct operation of your computer.

The backup Reference Diskette you created to hold the copy of option diskette contents (by selecting option 5 from the Main Menu) provides the ability to configure your PS/2. To do so, you can select option 5 from the Set Configuration menu (displayed in Figure 4.4)—Run automatic configuration. The selection of this option will result in the automatic configuration of the computer and any installed IBM options to their normal settings. Although you can also use option 2, Change configuration, to configure your system, this option requires you to know such technical details as adapter board ROM and RAM address ranges, which may not be readily available, but which are read from configuration files on the backup Reference Diskette when you select the automatic configuration option.

Option 3 from the Set Configuration menu, Backup configuration, causes the configuration in CMOS memory to be written onto the backup Reference Diskette. Then, if you should replace your computer's battery, you can use option 4, Restore configuration, to restore your computer's configuration to CMOS memory.

Set Features Menu

When you select the Set Features option from the Reference Diskette Main Menu, you can set the date and time, enter passwords, and set the speed to which the keyboard responds when you type. This menu is illustrated in Figure 4.7.

Figure 4.8 illustrates a portion of the display after you select the Set date and time option from the Set Features menu. Here the current date and time are displayed, and the cursor is positioned to the first entry in the date field, which is highlighted by an inverse video display bar. After entering a new date, you can use the Down Arrow key to position the highlighted bar on the time field and enter any required time changes. Then, pressing the Enter key updates the date and time in the computer's system clock.

Figure 4.7
Set Features Menu

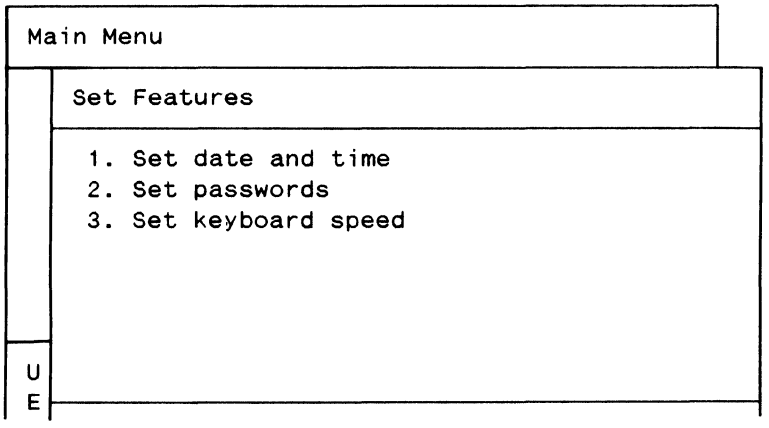
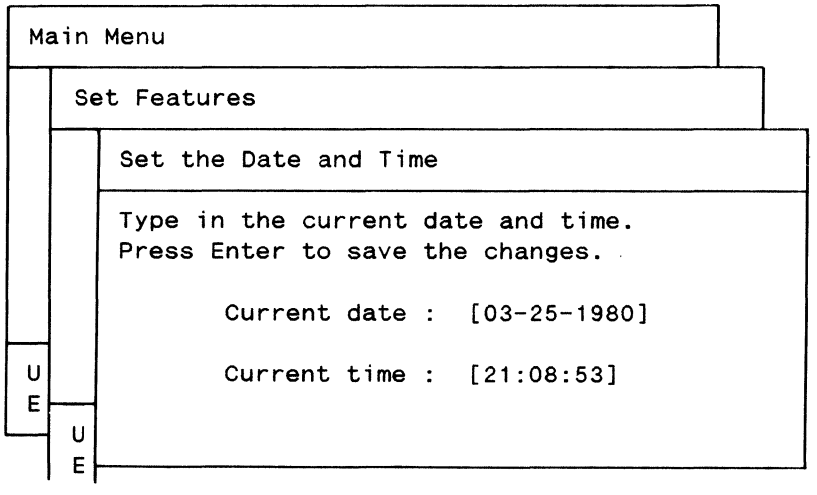


Figure 4.8
Set the Date and Time Menu



Select the Set passwords option from the Set Features menu to enter, change, or remove a power-on password. In actuality, the change power-on password option—invoked by selecting option 2 in the menu shown in Figure 4.7—is misleading, because once a password is set, you must turn your computer off and back on and await the display of the password prompt to be able to change it. Once a symbol that looks like a key is displayed in the upper left corner of your monitor, you can change the password by entering the current password followed by a slash and the new password. Selecting the change password option merely informs you of this procedure.

If you set a power-on password, each time you turn your computer on or perform a system reset, a key-shaped symbol appears to prompt you to enter a password to obtain control of the computer. If you should lose or forget the password, you will have to remove the battery in the system unit for at least 20 minutes to erase the CMOS memory containing your password. Then you must use the backup Reference Diskette to restore your system's configuration and set a new password if you so desire.

If you enter the password incorrectly, after three tries you will be precluded from further attempts to enter a password. At that time you will have to turn the computer off, then back on, and try again. When you enter the password incorrectly, the letter `X` displays under the key symbol, and entering the password correctly will cause `OK` to display under the key prompt.

Two other options in the Set Passwords menu are `Install keyboard password program` and `Set network server mode`. Setting a keyboard password enables you to lock the keyboard without turning the computer off. To set a keyboard password, you install the keyboard password program from the Reference Diskette onto the fixed disk or diskette that contains a copy of the OS. Because the fixed disk in a PS/2 must be partitioned and formatted before you can use it, prepare that disk and copy the operating system to it. After those functions are performed (as described in Chapter 5), you can use the Reference Diskette to set the keyboard password.

The last option from the Set Passwords menu—`Set network server mode`—enables the computer to operate as a local area network (LAN) server with its keyboard locked. This permits users of other computers to access the fixed disk of your computer while your keyboard remains locked.

Option 6 from the Main Menu, `Move the computer`, should always be selected before you move a system unit that contains a fixed disk, even if the move is only a few feet from one desktop to another. Selecting this option "parks" the read/write heads of the fixed disk in a location where, if the system unit is bumped, the heads will not alter previously recorded data. Once you invoke this option, you should not use the computer until it is relocated, as its further use "unsecures" the read/write heads.

Option 7 on the Main Menu (`Test the computer`) invokes a testing program. This program first displays a list of devices the program recognizes as being installed in your computer and then tests the various features installed in your system unit. To run this test program effectively, insert a scratch diskette in drive A. The program will first write test data to the diskette and then read the data to validate the accuracy of the diskette drive.

Media Compatibility Issues

The development of super-twisted and backlit LCD displays has greatly enhanced the ability of laptop computer users to read information displayed on laptop screens. Corresponding to the increase in screen readability, laptop computer sales have increased.

Although laptop computers are a valuable asset for the traveling professional, most computers in this category use 3½-inch media for removable disk storage. Unfortunately, the majority of IBM PC and compatible personal computers currently installed contain the older 5¼-inch media. Whereas the sale of members of the IBM PS/2 family of computers—which use 3½-inch diskettes—has been significant, problems also arise when you attempt to transfer information from those computers to laptop computers. In addition, a significant number of recently introduced IBM PC compatible computers, including the COMPAQ Deskpro 386, use 5¼-inch diskettes, resulting in media incompatibility problems that can inhibit the transfer of programs and data from a large base of personal computers to members of the PS/2 family. Knowing about diskette

capacities and the reasons media incompatibilities developed prepares you to master methods to obtain file transfer compatibility between laptop and desktop computers.

Types of Media

Currently four distinct types of diskettes are used in IBM PC and compatible computers and the PS/2 family of personal computers.

The original IBM PC series—the PC and PC XT—used 5¼-inch diskette drives that supported double density diskettes. Each diskette under DOS 2.0 and higher DOS releases could be formatted with 40 tracks per side, using either 8 or 9 sectors per track, with each sector containing 512 bytes of information. Thus, the maximum storage capacity of the standard 5¼-inch double density diskette used with the IBM PC and PC XT became

2 sides × 40 tracks/side × 9 sectors/track × 512 bytes/sector = 368,640 bytes

Because there are 1024 bytes in 1K byte, the storage capacity of the standard 5¼-inch diskette was 360K bytes of information.

When the IBM PC AT was introduced, it was marketed with one high-capacity 5¼-inch diskette drive as a standard feature. This diskette drive could read data previously recorded onto a standard 360K byte 5¼-inch floppy diskette. However, if data was recorded onto a 360K byte diskette in the PC AT's high-capacity drive, it was likely to be unreadable subsequently by a conventional 360K byte 5¼-inch diskette drive. The reason for this potential incompatibility between data recordings produced by different 5¼-inch diskette drives is based on the data recording mechanism employed by the high-capacity diskette drive used in the PC AT.

The 5¼-inch high-capacity diskette drive in its high-capacity recording mode of operation writes data onto a diskette using 80 tracks, with 15 sectors per track. Because the tracks are closer together, the high-capacity recording mode of operation requires high-capacity 5¼-inch diskettes. When data is recorded onto a high-capacity diskette, the maximum amount of storage becomes

2 sides × 80 tracks/side × 15 sectors/track × 512 bytes/sector = 1,228,800 bytes

Because there are 1024 bytes per 1K byte, the storage capacity of a 5¼-inch high-capacity diskette is 1.2M bytes.

When conventional 5¼-inch diskettes are used in the PC AT's high-capacity diskette drive and the operator formats the diskette as a 360K byte disk, the resulting tracks are placed slightly closer to one another than if a conventional 360K drive had been used to format a 360K diskette. Due to this, there is a degree of probability that data recorded onto a conventional 5¼-inch diskette in the high-capacity drive of the PC AT will not be readable in the standard 5¼-inch drive in IBM PCs and PC XTs. Thus, a high-capacity diskette produced by a PC AT is always incompatible with the 5¼-inch diskettes used in PCs and PC XTs, whereas a conventional 360K byte diskette produced by the high-capacity diskette drive of the PC AT may or may not be readable by the standard diskette drives used in the PC and PC XT.

As a result, IBM offered a standard 360K byte drive as an option that could be installed in the PC AT. The PC AT user then obtained the assurance of media com-

patibility with the standard 360K byte 5¼-inch diskette drives used in the PC and PC XT.

The introduction of the IBM PC Convertible laptop computer represented the first use by IBM of 3½-inch media in a personal computer. The PC Convertible's 3½-inch diskette drives record data onto a similarly sized diskette whose storage capacity is

2 sides × 80 tracks/side × 9 sectors/track × 512 bytes/sector = 737,280 bytes

With 1024 bytes per 1K byte of storage, the capacity of the 3½-inch diskettes used with the PC Convertible is 720K bytes.

The introduction of the IBM PS/2 family of computers included the support of two types of 3½-inch drives and media—double density 720K byte and high-capacity 1.44M byte storage. The IBM PS/2 Models 25 and 30 each use 3½-inch double density diskette drives that are the same as those used in the PC Convertible, having a storage capacity of 720K bytes. The PS/2 Models 50, 60, and 80 use high-capacity 3½-inch diskette drives that have a maximum capacity of 1.44M bytes of data. This doubling of storage capacity is obtained by recording data 18 sectors per track, using 512 bytes per sector.

File Transfer Methods

If you followed all the math just given, you see that there are four media formats you should consider for file transfers between a PS/2 and another PS/2 or between a PS/2 and a PC or PC compatible computer:

- 360K byte 5¼-inch
- 1.2M byte 5¼-inch
- 720K byte 3½-inch
- 1.44M byte 3½-inch

Thus you should take into account both the media format used by your computer and that of the computer to which you will transfer files to determine whether there is a media incompatibility problem. If there is, explore the variety of methods to obtain a file transfer compatibility between computers. In considering each method, note both the recording media used by both computers and the various types of facilities and equipment that can be employed to promote file transfers.

Table 4.1 lists six methods for transferring files between your computer and another computer. Of course, the implied seventh method, which is not listed in the table, is simply taking a diskette created on one computer and inserting it in a standard diskette drive on the PS/2.

Add Diskette Drive to Other Computer

The first method listed in the hardware column of Table 4.1 is the installation of a 3½-inch diskette drive on a non-PS/2 computer. This depends on the other computer's existing diskette configuration and the type of media used by that computer. If the other computer is an IBM PC series or compatible computer, an internal or external 3½-inch diskette drive can be added to the computer to obtain media compatibility with a PS/2. Several vendors market both types of 3½-inch drives, with internal drives obtainable for under $150 to include a mounting bracket to enable the drive to fit in

Table 4.1
File Transfer Methods

Hardware Methods	Software Methods
Install 3½-inch diskette drive in other computer	Use commercial file transfer program
Add diskette drive to PS/2	Use IBM Data Migration Facility
Use third computer as intermediate storage device	Use communications program on each computer

the device housing area of the computer that was designed for 5¼-inch storage devices. External drives containing a built-in power supply normally cost between $300 and $400 and must be cabled to the floppy diskette controller inside the system unit of the computer. For either type of drive, the desktop system user must obtain DOS 3.2 or a higher version of the operating system to support the use of 3½-inch diskette drives.

Add Diskette Drive to PS/2 Computer

If your computer is a member of the IBM PS/2 series, it has either a 3½-inch 720K byte or 1.44M byte diskette drive. The latter reads 720K byte diskettes and writes to those diskettes. If the other computer you wish to share files with cannot support 3½-inch diskettes, an IBM 5¼-inch external diskette drive can be cabled to your PS/2 to obtain media compatibility. Unfortunately, there are a few limitations and constraints associated with this device as well as its list price of approximately $400 that will make third-party products highly desirable. First, IBM's external diskette drive is probably the largest 5¼-inch drive ever manufactured, with many competitors only half in jest saying it's big enough to land a jet on. Because desk space can be at a premium in many organizations, its use may require some management of desk space. A second limitation is that when used with a PS/2, the 5¼-inch drive must be designated as drive B. Third and perhaps most important to users of the 3-expansion slot Models 50 and 70 computers is the fact that the 5¼-inch drive requires the use of an expansion slot for a separate controller, significantly reducing the expansion potential of those personal computers.

Commercial File Transfer Program

One of the most popular methods of file transfer is based on the use of the serial ports of the laptop and desktop computers, a cable to connect the two serial ports, and software that operates on both computers. Several vendors market a software/hardware package consisting of software on both 3½-inch and 5¼-inch diskettes and a cable with both 9-pin and 25-pin connectors on each end. By cabling the serial port of a PS/2 computer to the serial port of a PC or PC compatible computer and loading the vendor's software on each computer, you can transfer files in either direction.

One of the most popular commercial file transfer programs marketed today is called The Brooklyn Bridge. Developed by White Crane Systems of Norcross, Georgia, this package has a retail price of $129.95. In addition to permitting file transfers between

computers, the software also enables one computer user to access the other computer's peripheral devices. This additional feature enables a PS/2 user, for example, to use the printer or other peripheral attached to a PC or PC compatible computer through one simple serial port cable connection.

IBM Data Migration Facility

The IBM Data Migration Facility is a simple parallel cable connector and software that is designed to transfer files between a member of the PC series and a member of the PS/2 series.

The Data Migration Facility requires the cabling of two adjacent computers using the parallel port of each computer. Although this method of file transfer costs only $33, it is unidirectional from PC 5¼-inch drives to PS/2 3½-inch drives. Unfortunately, because the Data Migration Facility is designed for one-way file transfers, it cannot be used to send files from the PS/2's 3½-inch media to a PC's 5¼-inch media.

Using Communications Programs

If communications software programs are available for use on both a PS/2 and PC or PC compatible computer, a null modem can be used to cable the two computers together, using the serial port of each computer. Then the built-in file transfer capability of the communications software can be used to send data in either direction.

The cost of a null modem cable is normally under $20, whereas several communications programs with file transfer capability can be obtained for under $75 per copy. Thus, for $170 or less, you can achieve bidirectional file transfer capability between computers.

Third Computers as Intermediate Storage

If both computers have a communications capability, file transfer operations do not have to be restricted to occurring within close proximity of each computer. If the organization has another computer system, such as a minicomputer or mainframe computer with dial network access, one computer user can dial the intermediate storage computer and transfer a file onto that computer. Similarly, the other user can dial the intermediate storage computer and retrieve the previously stored file or transfer a file to that system, which later is uploaded to the other computer.

5 / Using DOS

For many PS/2 owners and users, IBM's disk operating system (DOS) provides a sufficient level of functionality and capability to forego using OS/2. For other PS/2 owners and users who will use OS/2, the commonality of a majority of commands and functions between that operating system and DOS permits persons familiar with DOS to ease themselves into the use of OS/2. For these reasons, this chapter and several succeeding chapters explore DOS. Then you'll be better prepared to understand OS/2, which is covered in detail later in this book.

Versions of DOS

The first year after the PS/2 family was introduced, DOS Version 3.3 was the primary operating system for this series of personal computers. In mid-1988, IBM introduced DOS 4.0, which contained many new and enhanced commands, as well as a menu system that facilitates its use. Because several million copies of DOS 3.3 were sold for use with PS/2 computers and this version of the operating system was being marketed concurrently with DOS 4.0 at the time this book was prepared, both versions are covered in this book. Due to the large degree of commonality between DOS 3.3 and DOS 4.0, this text first examines specific operating system topics using DOS 3.3. Then you will learn of relevant enhancements available in DOS 4.0. If a specific topic is the same for both versions of the operating system, this discussion does not specify a particular version of DOS.

Device Designators

Prior to using DOS, you should become familiar with the *designator* or *specifier* used for each storage device installed in or attached to your computer. This familiarity is required because the manner in which you start DOS depends on the storage devices installed in or attached to your computer.

If your computer has only one diskette drive, it will be referenced as physical drive specifier A. If your computer has two diskette drives, the drive installed in the left side of the system unit is referenced as drive A, whereas the drive installed in the right portion of the system unit is referenced as drive B.

If you have only one physical diskette drive, DOS treats that drive as logical diskette drives A and B. Doing so enables you to copy the contents of all or a portion of one diskette onto another, with DOS prompting you to change or "swap" diskettes.

If your computer has one diskette drive and one fixed disk, the diskette drive functions the same as if you had only one diskette drive. That is, the floppy drive is referred to by DOS as both drive A and drive B. The fixed disk is referred to as drive C.

Although most PS/2 computers have three or fewer drives, on occasion systems contain a second fixed disk. This disk drive is referenced as drive D. For media compatibility with members of the original IBM PC series, you may connect an external 5¼-inch diskette drive to your PS/2 computer. When this occurs, the drive specifier used to reference the external diskette drive normally is B. However, the drive can be varied based on the parameters you list in the DEVICE = DRIVER.SYS command contained in the CONFIG.SYS configuration file. The commands that can be contained in the CONFIG.SYS file are covered in Chapter 8.

Installing DOS 3.3

Your DOS 3.3 diskette contains a file named SELECT. This file can specify the keyboard layout you wish to use, the country code that will govern the format in which the date and time are displayed, and the currency symbol and decimal separator used by your computer. Using the SELECT file during the DOS 3.3 installation process, you automatically create a second copy of the operating system by responding to a few prompts issued by the command. The use of this command is applicable to all possible PS/2 configurations, regardless of the number of diskette drives or fixed disks your system contains.

If your PS/2 does not have a hard disk, installation requires one blank 3½-inch diskette, which becomes your "DOS Start-Up/Operating Diskette." If your PS/2 has a fixed disk, the SELECT command transfers the contents of your original DOS 3.3 diskette to your fixed disk. Once this occurs, you can start DOS from your fixed disk and eliminate the requirement to use the original or a backup copy of the DOS 3.3 diskette.

Diskette-Based Systems

Before you initiate the SELECT command on your DOS 3.3 diskette you should determine the country and keyboard codes to be used with that command. These codes will be entered as command parameters; their permissible values are listed in Table 5.1.

After inserting your original DOS 3.3 Start-Up/Operating Diskette in drive A, you can either press **Ctrl+Alt+Del** to start DOS if your system was previously powered-on, or you can simply turn on power to your computer. For either situation, you can ignore the date and time prompts by pressing the Enter key when DOS asks you to enter a new date and time. After DOS displays a copyright notice, the prompt `A>` displays. The character `A` signifies that DOS will use the diskette in drive A to process any file reference commands that are entered without a specified device name. This prompt is also known as the *default diskette prompt* or *default drive,* because DOS assumes that all file references without a drive specifier are to the drive indicated by the prompt.

Table 5.1
DOS Country and Keyboard Codes

Country	Country Code	Keyboard Code
Arabic	785	
Australia	061	US
Belgium	032	BE
Canada (Eng.)	001	US
Canada (Fr.)	002	CF
Denmark	045	DK
Finland	358	SU
France	033	FR
Germany	049	GR
Hebrew	972	
Italy	039	IT
Latin America	003	LA
Netherlands	031	NL
Norway	047	NO
Portugal	351	PO
Spain	034	SP
Sweden	046	SV
Switzerland (Fr.)	041	SF
Switzerland (Ger.)	041	SG
United Kingdom	044	UK
United States	001	US

Once A> appears on your display, you are ready to use the SELECT command, whose format is

SELECT *xxx yy*

where *xxx* is the country code and *yy* is the keyboard code with which you want to configure DOS to work. Assuming you wish to use the United States country and keyboard codes, from Table 5.1 you select 001 as the country code and US as the keyboard code. Then, you enter the SELECT command as follows:

SELECT 001 US

Figure 5.1 illustrates the screen display you should see as you start DOS 3.3 and enter the SELECT command. Note the warning message displayed after the SELECT command is entered. This message is given because the SELECT command invokes the DOS FORMAT command, which prepares a diskette or fixed disk for data recording. During this preparation, the FORMAT program writes marks on concentric circles that are used as indicators to position data as it is recorded to disk. These marks in effect erase any previously recorded data, so you are warned of this erasure in advance.

Figure 5.1
Using the SELECT Command

```
Current date is Thu  6-02-1988
Enter new date (mm-dd-yy):
Current time is 12:52:29.15
Enter new time:


The IBM Personal Computer DOS
Version 3.30 (C)Copyright International Business Machines Corp 1981, 1987
             (C)Copyright Microsoft Corp 1981, 1986

A>SELECT 001 US

SELECT is used to install DOS the first
time.  SELECT erases everything on the
specified target and then installs DOS.
Do you want to continue (Y/N)?  Y
```

Because the SELECT command automatically initiates the FORMAT command, you do not have to understand how to use the FORMAT command and its options at this time. Later this chapter reviews its use. FORMAT is extremely important, as it is the first step in preparing or initializing blank diskettes as you make duplicate copies of application programs or store or copy data files.

Assuming you wish to continue the SELECT command process DOS displays the character Y, so you simply press the **Enter** key to resume the operations initiated by the SELECT command. When this occurs you see the message:

```
Insert new diskette for drive B:
and strike ENTER when ready
```

If you only have one diskette drive, that physical drive will be used as logical drives A and B. Thus, the prompt `Insert new diskette for drive B:` in actuality tells you to remove the original DOS diskette from drive A and insert the new blank diskette in that drive.

If your computer has two diskette drives, insert the new diskette in physical drive B, the drive installed either in the right portion of the system unit or below the A drive.

Once you press the Enter key, DOS begins to format the target diskette, with the head and cylinder numbers continuously updated as the format operation progresses. After formatting is completed, the following message is displayed:

```
Format Complete
System transferred
```

The first line in the message indicates that the formatting process was completed and the disk is initialized for data recording. The second line refers to the fact that the SELECT command initiated the operation of the FORMAT command with a parameter that caused three system files to be transferred to the newly formatted diskette. By containing system files, the new diskette becomes "self-booting." This means that the diskette will contain files in predefined locations that are automatically loaded by a section of code in the computer's ROM. Once loaded, these files provide an interface between the user entering data from the keyboard and the operation of application programs. Later, this chapter examines the structure of the FORMAT command and how you can set its parameters to transfer system files to a newly formatted diskette.

After the format operation is completed and the system files are transferred, DOS displays three lines of statistics. The first line shows the total number of bytes of disk space; the second line indicates the number of bytes used by the system files. The third line displays the number of bytes available on the disk, which is the difference between the total disk space and the number of bytes used by the system files.

After displaying these statistics you see the message:

```
Format another (Y/N)?
```

When you enter the letter **N** (uppercase or lowercase), subsequent action depends on whether your computer has one or two diskette drives. If your system has one diskette drive, DOS prompts you when to insert the original DOS diskette and the `diskette for drive B:`. DOS displays `Reading source file(s)` when the original DOS diskette is inserted in drive A. After you insert your newly formatted diskette in physical drive A in response to the `Insert diskette for drive B:` prompt, the names of the files appear on your screen as they are copied. If your computer has two diskette drives, DOS automatically copies the files from the original DOS diskette onto the diskette installed in physical drive B without prompting you to `Insert a diskette for drive B:`. For either hardware configuration, the SELECT procedure is completed when the `A>` prompt is redisplayed. At this time, you should store your original DOS diskette in a safe place and use the recently created copy for everyday use.

Fixed Disk-Based Systems

If your computer system includes a fixed disk drive, you will probably want to install DOS 3.3 on that storage medium. By doing so you can subsequently load DOS from that storage device, so you will not have to insert a diskette in drive A each time you wish to use your computer.

Prior to installing DOS 3.3 on your computer's fixed disk, you must first prepare that storage medium to record data. To do so, use the DOS FDISK program. Insert the original DOS 3.3 Start-Up/Operating System diskette in drive A and power on your computer. Then, after the prompt `A>` is displayed, type **FDISK** and press **Enter**. The FDISK main menu is displayed, which is similar to the illustration shown in Figure 5.2. If your computer system has two fixed disks, the menu includes a fifth choice, which enables you to select the next fixed disk drive after you have prepared the first drive.

FDISK can create a *DOS partition,* which is an area on your fixed disk reserved for DOS to use. It stores your operating system files, as well as application programs designed to work under this operating system. Many PS/2 users require only one partition on their fixed disk; however, other users may require two or more partitions if they wish to install several operating systems on this storage media. Examples of other operating systems include a Microsoft implementation of AT&T's UNIX system, called XENIX, and CP/M-86, the latter an updated version of Digital Research's Control Program for Microcomputers (CP/M) operating system.

To create a partition for DOS 3.3, you accept the default choice of item 1 enclosed in brackets by pressing **Enter**. This action displays the screen illustrated in Figure 5.3.

As indicated by the options displayed in Figure 5.3, DOS has two DOS partition types. The first is called a *primary DOS partition* and is the only one required to use

Figure 5.2
FDISK Main Menu

```
IBM Personal Computer
Fixed Disk Setup Program Version 3.30
(C)Copyright IBM Corp. 1983,1987

FDISK Options

Current Fixed Disk Drive: 1

Choose one of the following:

    1. Create DOS Partition
    2. Change Active Partition
    3. Delete DOS Partition
    4. Display Partition Information



Enter choice: [1]



Press ESC to return to DOS
```

Figure 5.3
Creating a DOS Partition

```
Create DOS Partition

Current Fixed Disk Drive: 1

    1. Create primary DOS Partition
    2. Create extended DOS Partition

Enter choice: [1]

Press ESC to return to FDISK Options
```

DOS on a fixed disk. Under DOS 3.3 the maximum size of this partition was 32M bytes. However, several vendors of data storage devices offer extensions to DOS that break the 32M byte barrier. In addition, under DOS 4.0 the 32M byte partition limit was removed, with a partition size equal to the maximum disk space possible. If you do not have one of the third-party software extensions or use DOS 4.0, you can use FDISK to create up to three *extended DOS partitions.* These partitions can be any size and can be subdivided into multiple areas known as *logical drives,* with each logical drive limited in size to 32M bytes. Thus, if your fixed disk exceeds 32M bytes of storage capacity, under DOS 3.3 you would probably want to create an extended DOS partition and subdivide that partition into logical drives. Thereafter, when you load DOS 3.3, each logical drive is assigned a drive letter identifier you use to access the storage contained in the logical drive area.

To create a primary DOS partition, you select the default choice of 1 illustrated in Figure 5.3. This selection displays a new screen, illustrated in Figure 5.4. Normally, you want the primary DOS partition to be as large as possible if you do not intend to use another operating system. Thus, you would select the default choice of `Y` contained in brackets in Figure 5.4 by pressing **Enter.**

After the DOS partition operation is completed, the following message displays.

```
System will now restart

Insert DOS diskette in drive A:
Press any key when ready...
```

After you insert your DOS diskette in drive A and press a key, the "current" date and time are displayed, with DOS prompting you to enter a new date and time as previously illustrated in the top portion of Figure 5.1. Here the initial "current" date is a reference date and time when the original DOS Version 1.0 was produced and that you will obviously want to update to the true current date and time. Next, the copyright notice displays, after which the A> prompt appears. At this point you must use the SELECT command. This command sets your keyboard and country codes, formats your fixed disk, and transfers the files from the DOS diskette onto your fixed disk.

At the A> prompt you should enter the SELECT command using the following format:

```
SELECT C: XXX YY
```

Here C: is the drive specifier parameter that tells the command that it should operate on drive C. *XXX* is the country code, whereas *YY* is the keyboard code, with both codes selected from Table 5.1.

Once you enter the SELECT command, the same warning message as shown previously at the bottom of Figure 5.1 displays. If you continue the SELECT command process by pressing **Enter**, due to the severity of inadvertently formatting the fixed disk DOS displays

```
WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE C: WILL BE LOST!
PROCEED WITH FORMAT (Y/N)?
```

To proceed with the formatting operation, enter **Y**. During the format operation DOS updates the head and cylinder number each time they change to identify the progress of the format. DOS displays a message when formatting is complete and displays the message System transferred to denote that the three system files have transferred from the diskette onto the fixed disk. Next, DOS prompts you to enter a volume label:

```
Volume label (11 characters, ENTER for none)?
```

The volume label is normally used as an identifier for diskettes; however, it can also be used for fixed disks. Programs can be written to check the volume label to ensure

Figure 5.4
Creating a Primary DOS Partition

```
Create Primary DOS Partition

Current Fixed Disk Drive: 1

Do you wish to use the maximum size
for a DOS partition and make the DOS
partition active (Y/N).........? [Y]

Press ESC to return to FDISK Options
```

that a correct diskette is used or that the program is executed on a computer whose fixed disk was assigned a specific volume label. As indicated by the displayed message, you can either enter a volume label or simply press **Enter** to omit the label. After either action, disk space statistics display on your screen, followed by the message

```
Reading source file(s)...
```

As the remaining files on the DOS diskette are copied to your fixed disk, their names display on your screen. When all files have been copied, the A> prompt displays. At this point you have successfully installed DOS on your fixed disk and can remove the DOS diskette from drive A and store it in a safe place.

Installing DOS 4.0

Under DOS 4.0 the SELECT program was significantly enhanced. This program now operates as a full-screen utility that is automatically invoked when you power-on or press the Ctrl+Alt+Del keys to perform a system reset with the DOS 4.0 INSTALL diskette in drive A. Prior to installing DOS 4.0 you should ensure that you have at least one available blank diskette if your computer has 3½-inch high-capacity diskette drives or two blank diskettes if your computer has standard-capacity 3½-inch diskette drives.

Once you power-on your computer or perform a system reset with the INSTALL diskette in drive A the IBM logo and the program name DOS SELECT that is initiated will be displayed. This will be followed by a copyright notice and instructions to press the Enter key to continue or the Esc key to cancel the program. As the SELECT program operates, it displays the action that occurs and provides you with the ability to display "Help" information by pressing the F1 key as well as prompting you when to insert diskettes.

Diskette-Based System

If your PS/2 does not have a fixed disk drive, you will require two blank 3½-inch 720K byte diskettes. On one diskette the SELECT program places the DOS command files and utility programs. On the second diskette the SELECT program copies the DOS SHELL program and additional DOS utility programs. Thus, label the first diskette "Start-up" and the second, "Shell." Once the appropriate information is copied to the two diskettes, you can start DOS using either diskette. If you use the Start-up diskette, DOS 4.0 will be brought up in a command-based interface; whereas, if you use the Shell diskette, DOS 4.0 is initiated using the DOS Shell.

The DOS Shell is a graphics-based interface on PS/2 computers, and it presents a user-friendly display of available selections and incorporates an on-line help facility. Later sections of this chapter and succeeding chapters examine the operation and use of the DOS 4.0 Shell. Because the actual installation of DOS 4.0 is very similar for diskette and fixed disk operations, its use is described on a fixed disk system with notations about relevant differences between the fixed disk and diskette installation.

Fixed Disk System Installation

After the IBM logo screen is displayed, the SELECT program will display a screen of information denoting the number of blank diskettes you should have based on the storage

capacity of the diskette drives installed in your computer. In this display screen the term *1M diskette* refers to diskettes that have a formatted storage capacity of 720K bytes, whereas the term *2M diskette* refers to diskettes that have a 1.44M byte formatted storage capacity. This screen is illustrated in Figure 5.5. As mentioned earlier in the chapter, if your PS/2 only has diskettes with a 720K byte storage capacity, you will need two 3½-inch 1M byte diskettes, whereas systems with one or more 1.44M byte storage capacity diskette drives only require one blank 2M byte diskette. Finally, as indicated in Figure 5.5, if your PS/2 has a fixed disk you will only require one blank diskette.

After the information concerning the number of required blank diskettes is displayed, you specify a division between DOS functionality and program workspace. A new screen containing three entries displays; it is illustrated in Figure 5.6.

The second entry in Figure 5.6 is highlighted as the default value, which is selected if you press Enter. Essentially, this screen helps you to specify the amount of memory-resident DOS functions. Because DOS supports a maximum of 640K bytes, without invoking the DOS 4.0 support of expanded memory that requires programs specifically written for this feature, maximizing DOS functionality reduces your program workspace to a minimum value. Unless you intend to execute very large spreadsheet programs, selecting option 2 is acceptable for most users and should be selected if your computer's RAM is 512K bytes. If you expect to perform operations on large spreadsheets, select option 1 only if you are using a computer other than a PS/2 that has 256K bytes of RAM. For PS/2 computers with more than 512K bytes of RAM—all PS/2s except the Model 25—select option 3. This option maximizes DOS functionality while permitting sufficient memory to execute programs.

Once you select the program workspace, the next DOS 4.0 SELECT screen sets the country and keyboard parameters. Unlike DOS 3.3, which requires you to enter codes, under DOS 4.0 the U.S. country and keyboard codes are predefined and accepted if you simply press Enter. For a different country and keyboard, the DOS 4.0 SELECT

Figure 5.5
SELECT Program Diskette Requirements

```
                              Welcome

Welcome to DOS 4.00 and the SELECT program.  SELECT
will install DOS 4.00 on your fixed disk or diskette.
If you install DOS 4.00 on a diskette, the number of
blank diskettes you need depends on the type and
capacity of your diskette drive:

   Drive Type (Capacity)       Number of Diskettes

   5.25-Inch Drive (360KB)     four 5.25 (360KB)
   5.25-Inch Drive (1.2MB)     four 5.25 (360KB)
   3.5-Inch Drive (720KB)      two  3.5  (1MB)
   3.5-Inch Drive (1.44MB)     one  3.5  (2MB)

If you install DOS 4.00 onto a fixed disk, you need
one blank diskette:

   5.25-Inch Drive             one  5.25 (360KB)
   3.5-Inch Drive              one  3.5  (1 or 2MB)

    Press Enter (↵) to continue or Esc to Cancel

Enter    Esc=Cancel
```

Figure 5.6
SELECT Function and Workspace Menu

```
                    Specify Function and Workspace

SELECT sets up your computer to run DOS and your programs
most efficiently based on the option you choose.

Note:  You can review the results of your choice later
in this program.

Choose an option:

    1. Minimum DOS function; maximum program workspace

    2. Balance DOS function with program workspace

    3. Maximum DOS function; minimum program workspace

Enter     Esc=Cancel     F1=Help
```

Figure 5.7
DOS 4.0 Country Selection

```
                          Country Selection

Choose a country:

    United States              (001)    Norway                     (047)
    Canada (French Speaking)   (002)    Germany                    (049)
    Latin America              (003)    Australia                  (061)
    Netherlands                (031)    Japan                      (081)
    Belgium                    (032)    Korea                      (082)
    France                     (033)    Peoples Republic of China  (086)
    Spain                      (034)    Taiwan                     (088)
    Italy                      (039)    Portugal                   (351)
    Switzerland                (041)    Finland                    (358)
    United Kingdom             (044)    Arabic Speaking            (785)
    Denmark                    (045)    Hebrew Speaking            (972)
    Sweden                     (046)

Enter     Esc=Cancel     F1=Help
```

program displays the available countries and their codes, as illustrated in Figure 5.7. Initially, a highlighted bar displays over the United States entry on the screen. You can use the Up and Down arrows to reposition the highlight bar to the required country and press the Enter key to make your selection. In comparison, under DOS 3.3 you must first look up the appropriate country code and then enter its numeric value.

Once you complete the country and keyboard selections, the Select Installation Drive menu is displayed. This menu provides the capability to specify the drive on which DOS will be installed. If you have a fixed disk, one option shows the drive designator C, whereas a diskette-based system shows drive A for this option.

After you specify the drive on which to install DOS, the next menu contains options to specify a location in a hierarchical directory structure to which DOS files will be copied. Essentially, DOS supports a directory structure similar to an inverted tree, with the root directory denoted by the backslash character (\) at the top of the structure. Paths to subdirectories begin with a backslash, followed by the name or names of the subdirectories in the route to a specific subdirectory, with a backslash preceding each name.

Figure 5.8 shows the Specify DOS Location menu. This menu actually provides several other functions as well. First, as specified by the name of the menu, you can accept the default displayed location \DOS, which defines a subdirectory under the root directory on drive C that will be named DOS as the location where files will be copied from the INSTALL diskette. For most PS/2 users, this is acceptable; however, to locate DOS files elsewhere you can enter a string of up to 63 characters between the brackets shown in Figure 5.8. This is the maximum path length supported by DOS. If you enter a DOS location, the SELECT program creates a subdirectory to match your entry if a previously created directory does not exist.

As indicated in Figure 5.8, you can use the Specify DOS Location menu to have all DOS files on a fixed disk updated. This feature enables you, as an example, to update DOS 3.3 to DOS 4.0. If you select option 2, you can have all nonsystem files copied to a specified directory whose name you enter in brackets.

Once DOS files are copied to the default or a specified subdirectory, the SELECT program displays several menus requesting information concerning the number and type of printers you have and the computer port(s) to which they are connected. From anywhere on these menus you can press the F1 key to obtain on-line assistance; a sample help window appears in Figure 5.9. Note in the figure that a "Help" message is superimposed over the Printer Selection menu as a result of pressing the F1 key.

Once appropriate printer information is entered, SELECT can accept the previously entered data and continue with the installation, or it can review, change, or add in-

Figure 5.8
DOS 4.0 Specify DOS Location Menu

```
                        Specify DOS Location

You can accept the DOS directory name shown or type a new
directory name.

DOS Directory . . . .C:\[DOS                                    ]


To select option 1 below, press Enter.  To change your
option, press the tab key, highlight your choice and then
press Enter.

    1. Update all DOS files on fixed disk

    2. Copy non-system files to directory specified


Enter     Esc=Cancel     F1=Help
```

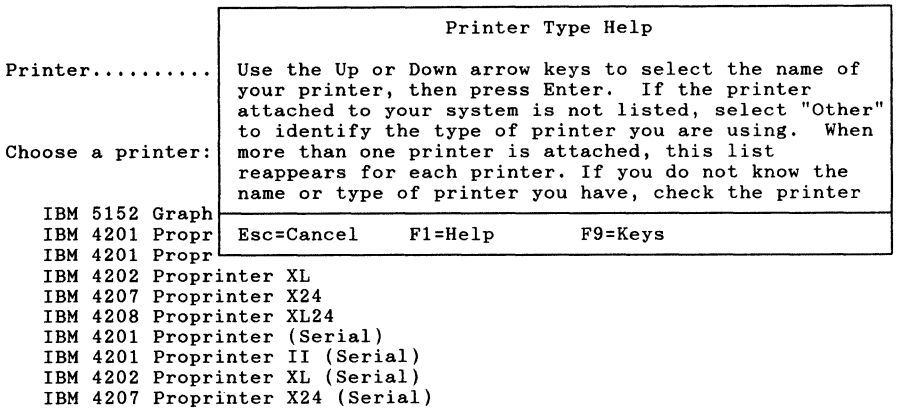


Figure 5.9
Help Message Window Displayed in Front of the Printer Selection Menu

stallation choices. Assuming you accept the previously entered data, SELECT makes a copy of the INSTALL diskette and prompts you when to remove that diskette and to insert your backup diskette. After the copy of the INSTALL disk is made, SELECT copies the contents of that diskette and the OPERATING diskette to the fixed disk if you previously installed DOS 3.3 on the fixed disk. If this is the first time you are using the fixed disk, SELECT displays a menu that you use to partition the disk. Under DOS 4.0 partitions can exceed 32M bytes, so you can either let SELECT define the partition size—which sets it to the maximum storage capacity of the disk—or you can define the partition size.

After the partition size is defined, you are instructed to perform a system reset operation. Press the **Ctrl+Alt+Del** keys. At this time SELECT formats your fixed disk. During formatting, SELECT displays the percentage of the disk that is formatted in the upper left corner of the screen. Once formatting is complete, SELECT copies the files from the install diskette to the subdirectory \DOS or another previously specified subdirectory. Then you are instructed to insert the Operating diskette, whose contents are now copied to the fixed disk. After the two copying operations are completed, SELECT prompts you to remove all diskettes and perform another system reset operation. This indicates that the DOS 4.0 installation process is completed, and a system reset brings up the DOS 4.0 Shell.

DOS Shell

Figure 5.10 illustrates the initial DOS 4.0 Shell menu, whose title is Start Programs. If you check the upper left corner of Figure 5.10, notice an arrow; it is the *mouse pointer*. The DOS 4.0 Shell supports both keyboard entries and the use of a mouse, whereas DOS 3.3 is limited to keyboard entries.

The three items listed under the date in the left portion of the Start Programs display—`Program`, `Group`, and `Exit`—are referred to as items on the *action bar*. By pressing the F10 key, you can use the action bar. Pressing this key highlights the Program entry on the action bar. This highlighting feature is the *selection cursor*, and it can be moved to other items in the action bar by pressing the Left and Right arrow keys.

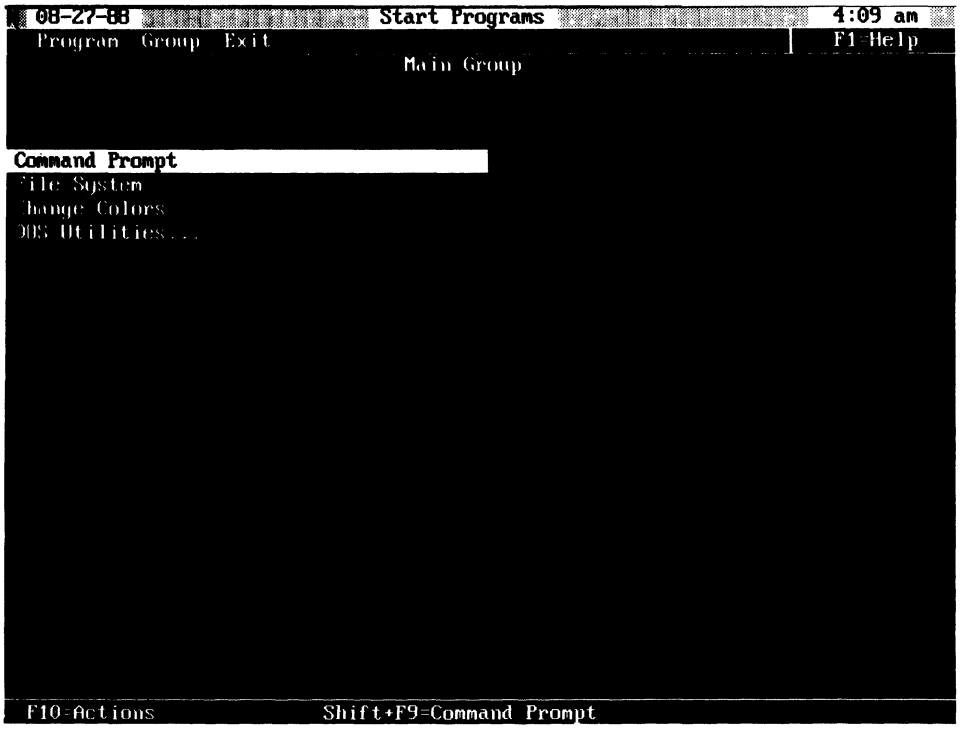


Figure 5.10
DOS 4.0 Main Group

As an alternative to moving the selection cursor over an appropriate item, you can press the character key that appears underlined in the selection. As an example, you can type the character **G** to select Group or the character **X** to select Exit. Once you press the Enter key, a pull-down menu listing the options for the item in the action bar is displayed. Figure 5.11 illustrates the resulting display of the Program pull-down menu. As this chapter examines the operation and usage of DOS commands, it also investigates the use of the DOS Shell when appropriate. For now, look at the four entries in the vertical column in the left of the screen. These entries, starting with `Command Prompt`, are located in the Shell's group area. When the DOS Shell is initially displayed, the `Command Prompt` entry is highlighted. Using the Up and Down arrow keys you can reposition the selection cursor to other items in the group area. Pressing Enter when the Command Prompt item is highlighted switches DOS 4.0 to its command prompt mode of operation, similar to DOS 3.3. Figure 5.12 shows the message you see after you switch to this mode. Note that you can use command prompt mode anytime in the DOS Shell by pressing the key combination **Shift+F9**. You can type **EXIT** to return to the DOS Shell.

Also note that the default prompt used by DOS 4.0 is `C:\DOS>`, instead of `C>` for a fixed disk system using DOS 3.3. This change is caused by the INSTALL program, which creates a file called AUTOEXEC.BAT whose DOS PROMPT command alters the prompt. The AUTOEXEC.BAT file is automatically executed whenever you power-on your computer or perform a system reset operation, causing the PROMPT command

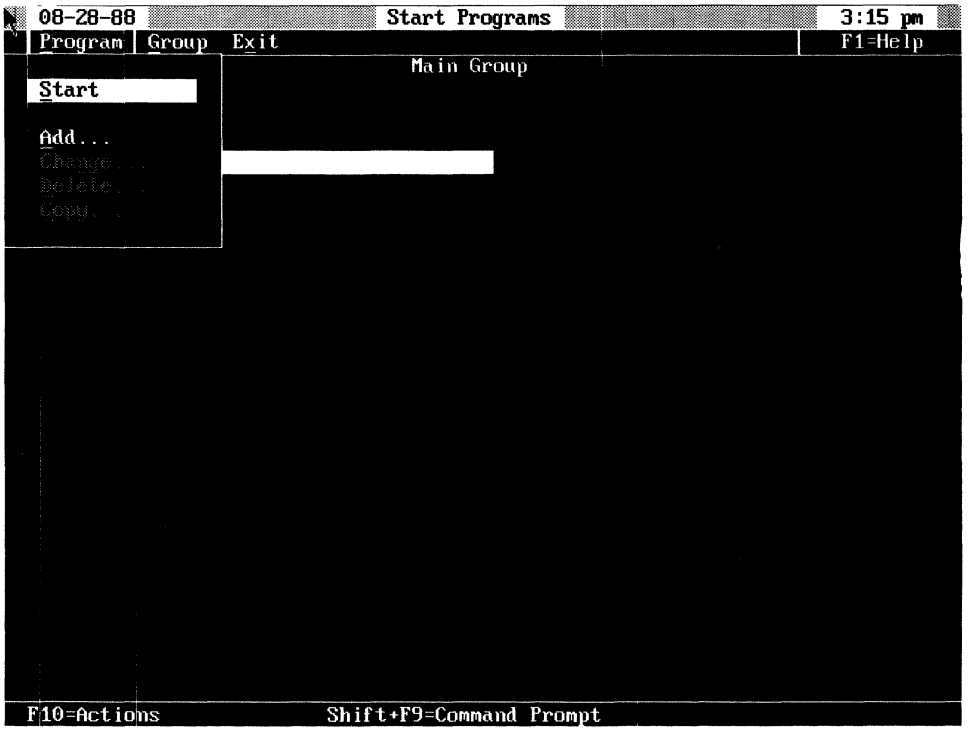


Figure 5.11
DOS 4.0 Start Programs Display

```
When ready to return to the DOS Shell, type EXIT then press enter.

IBM DOS Version 4.00
         (C)Copyright International Business Machines Corp 1981, 1988
         (C)Copyright Microsoft Corp 1981-1986

C:\DOS>
```

Figure 5.12
Initialized DOS 4.0 Command Mode

in the file to be changed from the DOS 3.3 default. If you are more comfortable with the previous prompt display, remove the PROMPT command from the AUTOEXEC.BAT file. Further information concerning the use of batch files to include the AUTOEXEC.BAT file appears in Chapter 8.

Bringing Up DOS from Drive A

Now that you have either made a duplicate DOS diskette or installed DOS on your fixed disk, you are ready to start the operating system for everyday use.

There are two methods that you can use to start DOS. If power to the system unit is off, you can insert your DOS diskette in drive A. Then, after power is turned on, the internal power-on self-test (POST) is performed, and DOS is automatically loaded into the computer's memory. If your system was previously powered-on, you can perform

a system reset operation (by pressing **Ctrl+Alt+Del**) to load DOS. When this operation is initiated, the computer clears its memory and restarts itself without performing a POST operation.

When either of the two previously discussed methods of starting DOS are performed, the computer's Basic Input/Output System (BIOS) in read only memory (ROM) causes any disk in drive A to be searched. If the disk contains DOS system files, the computer's BIOS loads three files into memory. The first two files are named IBMBIO.COM and IBMDOS.COM. These files are *hidden files,* because they are not listed if you list a directory of the disk where the programs reside. The first file provides a standard interface to the hardware and supplements the BIOS contained in the computer's ROM. The second file is responsible for interpreting commands issued by application programs and converting those commands into a form recognizable by BIOS. The third file is called COMMAND.COM, which is a command processor that accepts and processes DOS commands you enter from the keyboard or from a *batch file.* A batch file contains a frequently used sequence of DOS commands.

After the three core DOS files are loaded, the current date is displayed if you are using DOS 3.3, and you are prompted to enter a new date or accept the displayed date. If you are using DOS 4.0, the DOS Shell is loaded and the date and time are displayed at the top left and right corners of the screen (refer to Figure 5.10). For DOS 3.3, the current date is displayed, followed by the prompt shown here.

```
Current date is Fri 5-9-1989
Enter new date:
```

You can enter any month, day, and year as long as they fall within the following ranges:

month (m) is 1 or 2 digits from 1 to 12
day (d) is 1 or 2 digits from 1 to 31
year (y) is 2 digits from 80 to 99 or 4 digits from 1980 to 2099

The delimiters between the month, day, and year can be either a slash (/) or hyphen (-). If you enter an invalid date or delimiter, DOS repeats the message, as in

```
Enter new date: 12-23/89
Enter new date: 12-23-89
```

until the format is correct.

After an acceptable date is entered, DOS 3.3 displays a message similar to

```
Current time is 0:01:21.85
Enter new time:
```

Note that under DOS 3.3 the time is displayed as

hours:minutes:seconds.hundredths of a second

whereas only hours and minutes are displayed when the DOS 4.0 Shell is used. You can enter any time, as long as it falls within the following ranges:

hours is 1 or 2 digits between 0 and 23
minutes is 1 or 2 digits between 0 and 59
seconds is 1 or 2 digits between 0 and 59
hundredths of a second is 1 or 2 digits between 0 and 99

When you enter the time, be sure to enter a colon (:) after each time element (except hundredths of a second) for which you must enter a period—a slash (/) or a hyphen (-) will not work.

Time must be entered in a military format, with 1 p.m. expressed as 13 hours, 2 p.m. expressed as 14 hours, and so on. Due to this, be sure to convert the appropriate hour of the day to its correct military format. Also note that you can press the Enter key without entering a time to accept the displayed time, or you can enter just the hour or however much of the remaining levels of time information you wish.

Once you enter the date and time or respond to their prompts by pressing Enter, a copyright notice appears. Then the prompt A> displays, indicating that the diskette in drive A will be examined automatically to process any file reference commands that you enter without a specified device name. The complete DOS 3.3 initialization procedure for a diskette-based system is illustrated in Figure 5.13.

If you initialize DOS 4.0, the DOS Shell display is similar to that in Figure 5.10, with the icon for drive A highlighted instead of the icon for drive C. Then, if you press the Shift+F9 keys or select the Command Prompt option, you switch to the operating system's command mode of operation, with a display similar to that shown in Figure 5.12. The only difference for diskette initialization is that the prompt under DOS 4.0 is A:\> instead of C:\DOS> when the operating system is initialized from the fixed disk.

Bringing Up DOS from Drive C

If your PS/2 has a fixed disk, you will normally initialize DOS from that device. By doing so you forego placing a DOS diskette in drive A when you power-on your computer or perform a system reset operation.

If you previously installed DOS on drive C, that drive will be automatically searched for DOS when you power-on your computer or perform a system reset operation. The DOS initialization procedure for a system with a fixed disk is similar to that illustrated in Figure 5.13 when using DOS 3.3, with the only difference being that the default prompt will become C>, indicating that the fixed disk is the default drive. As previously

Figure 5.13 DOS 3.3 Initialization on Diskette-Based System

```
Current date is Wed  6-15-1988
Enter new date (mm-dd-yy):
Current time is 11:00:38.08
Enter new time:


The IBM Personal Computer DOS
Version 3.30 (C)Copyright International Business Machines Corp 1981, 1987
             (C)Copyright Microsoft Corp 1981, 1986

A>
```

explained, if you initialize DOS 4.0 from a fixed disk and select its command mode of operation, the prompt C:\DOS> is displayed.

Changing the Default Drive

You can change the DOS default drive designation prompt by entering a new drive designation letter followed by a colon. The following examples illustrate the resulting prompts when you change the default drive under DOS 3.3 and DOS 4.0.

DOS 3.3	DOS 4.0
C>	C:\DOS>
C>**A:**	C:\DOS>**A:**
A>	A:\>

Here, the original prompt in the first row is changed when you enter **A:** (shown in the second row), becoming a drive A prompt in the third row.

The key difference between the drive designation prompt for DOS 3.3 and 4.0 is that under DOS 4.0 the current directory is also displayed. Thus, C:\DOS> indicates that drive C is the default drive and the subdirectory DOS located under the root directory is the current directory. Similarly, A:\ indicates that drive A is the default drive and the root directory (\) is the current directory.

As a result of entering A followed by a colon, the default drive was changed to drive A. Now, A will be the drive DOS will search for any commands or file names you enter. Note that for PS/2s that have only one diskette drive, changing the drive designation from A to B or from B to A has no effect on the physical drive that will be searched for commands or file names. This is because the one physical diskette drive will function as two logical drives in tandem with each drive designation change.

Editing Keys

To help you enter and modify commands, DOS assigns predefined editing functions to the first five function keys (F1 through F5), the insert (Ins), delete (Del), and escape (Esc) keys. In addition, the Backspace key can also be used for editing, eliminating one character to the left of the cursor each time that key is pressed.

Table 5.2 summarizes DOS editing keys and the functions associated with the use of each key. Note that no cursor control keys are listed in Table 5.2. This is because the normal cursor control keys are disabled when the computer is in the DOS command mode. Thus, you cannot move the cursor to a specific character location in a command to make corrections, unfortunately resulting in a rather primitive editing facility. However, you can use the Left and Right arrow keys to delete and redisplay one character at a time from the command line. The Left arrow key functions like the previously described Backspace key, and the Right arrow key functions the same as the F1 key.

As data is entered from the keyboard it is placed into a temporary storage area known as an *input buffer*. Data remains in this buffer until you press Enter, after which the keystroke is processed. Due to this, you actually modify or repeat the contents of the input buffer when you edit a command line.

Table 5.2
DOS Editing Keys

Editing Key	Function Performed
DEL	Deletes one character in the input buffer without moving the cursor.
Ins	Inserts characters.
Esc	Cancels the line currently displayed while the contents of the input buffer remain unchanged.
F1	Redisplays one character from the input buffer.
F2	Redisplays all characters up to a specific character.
F3	Redisplays all remaining characters from the input buffer.
F4	Skips over all characters in the input buffer until a specified character is encountered. This is the opposite of F2.
F5	Accepts the edited line for further editing with the currently displayed line being placed in the input buffer.

To illustrate the use of DOS editing keys, assume that in response to the DOS prompt `C>` or `C:\DOS>` you typed the following command line without pressing the Enter key.

DIR A:STAT.BAS

In the preceding command line the actual command invoked is the DOS directory command whose name is DIR. Here the DIR command operates on the contents of drive A, indicated by the letter A followed by a colon (:), and the file, whose name is STAT with the extension .BAS. Entering this command displays information concerning the file, including its size and date of creation. Everything to the right of the command (DIR) is called the *file specification.* The actual construction of file specifications will be covered later in this chapter.

After you type the previously mentioned command line, use the Backspace key to erase that line from the display. Although the data is erased from the screen, the line is still in the input buffer.

Each time you press the F1 key, one character from the buffer is copied to the screen. Thus, pressing the F1 key twice results in the following screen display:

```
DI
```

The F2 key performs a function that is similar to the multiple use of the F1 key. That is, pressing the F2 key followed by a single character that functions as a delimiter results in the display of all characters from the input buffer up to but not including the first occurrence of the delimiter. As an example of the use of F2, press that key and type an S. The screen appears as follows:

```
DIR A:S
```

You can use the F3 key to copy all of the remaining characters from the input buffer onto the screen. Once the Enter key is pressed, only the characters on the screen are sent to the computer for processing. To illustrate the usefulness of the F3 key, suppose

you wanted to check for the status of the file STAT.BAT on the diskette in drive A. By pressing F3, you would generate the display of the contents of the input buffer as follows:

```
DIR A:STAT.BAS
```

Now, you could use the Backspace key to erase the *S* character. Then you could type **T**, then press **Enter** to invoke the required operation. This would now provide a directory listing of the file STAT.BAT on drive A by changing one letter instead of typing a new command line. As you work with DOS editing keys you will find the F3 key to be most useful. This key can generate a previously entered command line. That line can be duplicated as much as you want, or you can modify the previously entered command before you send the command line to the computer for processing.

Like the F2 key, to use the F4 key you insert a delimiter. Here the delimiter informs DOS to skip all characters up to the first occurrence of the delimiter character when DOS displays the remainder of the contents of the input buffer. Note that if the specified character is not present in the input buffer, no characters are skipped.

To illustrate the use of the F4 key, assume your input buffer appears on your screen as follows:

```
DIR A:STAT.BAS
```

To list a directory of all files on the B drive whose extension is .BAS, either you could enter the appropriate command from scratch or you could take advantage of the commonality of one or more portions of the current input buffer using DOS editing keys. Here the command to list all files on drive B with the extension .BAS is

DIR B:*.BAS

The asterisk is a global filename character whose use is covered later in this chapter. If you press the F3 key, the contents of the input buffer are displayed, which were previously entered as

```
DIR A:STAT.BAS
```

By pressing the **Backspace** key 10 times or holding the key down, you remove all characters up to and including the drive designator A. Now you can type the three characters **B:***. Next, press the **F4** key followed by a period, which causes all characters up to but not including the period to be skipped. Then, pressing the **F3** key causes the remainder of the input buffer to be copied to the screen as illustrated below.

```
DIR B:*.BAS
```

Insert/Delete

The insert (Ins) and delete (Del) keys, despite operating as their names imply, actually work on the contents of the input buffer when you are at the DOS command level. Thus, their usefulness for command line editing is minimal.

As an example of the use of the Ins and Del keys, assume you want to check the status of the file FRED.BAS on drive B. The contents of the input buffer are

```
DIR B:*.BAS
```

You first press **Backspace** five times or hold that key down until the period is removed from the display. Next, press the **Del** key to erase the asterisk from the input buffer. Then press **Ins** to switch DOS to insert mode. Now you type the word **FRED**, causing each character to be inserted into the input buffer while all characters to the right of and including the period are shifted to the right. When you finish inserting the filename FRED, you press **Ins** key a second time to leave the insert mode. Then you press the **F3** key, causing the remainder of the command line to be displayed as in

```
DIR B:FRED.BAS
```

Now you can either press the Enter key to make this revision replace the data in the input buffer as well as send it to the computer for processing, or you can press the **F5** key to cause the contents of the displayed line to replace the contents of the input buffer for further editing. However, this does not send the command line to the system for processing. To distinguish this, DOS displays the @ character at the end of the line when the F5 key is pressed, after which the cursor is moved to the first position of the next line.

Control Functions

When you load DOS you can initiate five predefined control functions, based on the use of multikey combinations. These combinations and their operations under DOS are summarized in Table 5.3.

The system reset function is invoked when you simultaneously press Ctrl+Alt+Del. This process reinitializes DOS, which is useful if your system freezes due to a bug in an application program or some other abnormality occurs.

If you are using DOS 3.3, the Shift+PrtSc key combination prints the text contents of the display. If graphics are displayed, you can print them as well with Shift+PrtSc if a special DOS file named GRAPHICS was previously loaded and your computer is using the CGA display mode. Under DOS 4.0, the GRAPHICS program can print the

Table 5.3
DOS Multikey Functions

Key Combination	Function Performed	Description
Ctrl+Alt+Del	System Reset	Causes DOS to reload from the diskette or fixed disks.
Shift+PrtSc	Print Screen	Causes all data on the screen to be printed.
Ctrl+PrtSc	Echo to Printer	Causes all input and output to be logged to the printer.
Ctrl+NumLock	Suspend System Operation	Freezes the operation of the computer until a key is pressed.
Ctrl+Break	Break	Cancels current operation and returns to DOS prompt level.

contents of a screen that contains text and/or graphics in the CGA, EGA, MCGA, and VGA video display modes.

The Ctrl+PrtSc key combination can be viewed as a logging facility. When this key combination is first pressed, it prints whatever you type and all system responses. This echoing of data to the printer continues until you press this pair of keys again. This multikey combination is normally used to obtain a hardcopy historical log of operational procedures, error messages, command use, and system responses. Then the log can be examined to determine the possible cause of unexpected events, used as a reference for creating or debugging batch files, or stored for future reference.

Because in its normal video mode your computer screen can only display 25 lines of data, many times the listing of a disk's directory or a long file causes data to scroll rapidly off the screen. To freeze the screen after a certain number of lines, press **Ctrl+NumLock**. This key combination suspends the operation of your computer, freezing the display and enabling you to pause to think about your operation before you execute it. To resume operations, just press any key. To stop a previously suspended command, you can press **Ctrl+Break**. In fact, Ctrl+Break can be used to terminate the entry of a command line and return you to the DOS prompt level, as illustrated by the following example:

```
C>DIR B:FRE

C>
```

When you press Ctrl+Break after typing FRE, no directory listing is generated.

Command Syntax (Format)

A common method to describe the parameters that can be included in each DOS command is presented in this section. This format notation is used in the remainder of this book to identify the basic format of DOS commands as well as to denote the parameters that can be contained in a command line entry. Common format notations used for DOS commands include

- Keywords. Capital letters are used to identify DOS commands. Although keywords are shown here in capital letters, in actuality, any combination of uppercase and lowercase characters can be used.
- Command Parameters. Items shown in lowercase italic letters are command parameters. You supply these items when you enter the command.
- Optional Parameters. Items enclosed in square brackets ([ ]) are optional. You may or may not include them in a command.
- Repeating Items. Items that may be repeated as many times as you want are indicated by ellipses (. . .).
- Choose an Item. Items contained in braces ({ }) indicate you should select one item from the group.

Note that with the exception of square brackets, braces, and ellipses, all punctuation characters, such as commas, equal signs, and slashes, must be included as indicated in the command format.

Command Parameters

Drive Letter

The drive letter (*d*) followed by a colon is used to specify the drive that contains the command in the form of a file or the drive(s) the command will operate on.

If the drive letter precedes the command, DOS searches the indicated drive for a file containing the command to be acted on. For example, assume the default drive is A and you wish to execute the FORMAT command that resides on drive C. You would then enter this command at the A> prompt:

C:FORMAT

Under DOS 4.0 the actual location of the FORMAT command is under the subdirectory \DOS on drive C if you accepted the default location when the SELECT program was executed. Although you could enter the command to include its path as C:\DOS\FORMAT, you can omit the path and simply enter the command as **C:FORMAT**. The reason why you can enter the command without the path \DOS is that the SELECT program creates a batch file named AUTOEXEC.BAT that contains the DOS command PATH C:\DOS. The AUTOEXEC.BAT file is automatically executed each time you power-on your computer or perform a system reset. The PATH command placed in that file by the SELECT program causes DOS to search the subdirectory \DOS for programs and files not found in the current directory. Specific information concerning the PATH command and the use of batch files is part of Chapter 6.

If a drive letter does not follow the FORMAT command, that command will assume that a diskette in the default drive (A) is to be formatted. Thus, if you want to format a diskette in drive B using the FORMAT command resident on drive C, you enter the following:

A>C:FORMAT B:

Note that under DOS 4.0, the prompt A:\> is displayed, whereas under DOS 3.3 the prompt is A>.

When you power-on your computer, the initial default drive depends on your system's hardware configuration. If you do not have a fixed disk, when DOS is initialized the prompt A> or A:\> indicates that A is the default drive. If your computer system has a fixed disk on which DOS is installed, the prompt C> or C:\DOS> is displayed if you power-on your computer without a diskette in drive A. As indicated earlier in this chapter, you can change the default drive by entering a new designation letter followed by a colon.

Path

A path (*path*) is used in a tree-structured directory to specify the route to a file. The path follows the drive letter and precedes the filename.

Similar to the drive letter, the path to a file can be specified twice in most command lines. If the path follows the drive letter but precedes the command name, it indicates the route to the file on the drive that contains the command. If the path follows the

command and drive letter, it identifies the route to the file the command will operate on. A preliminary discussion of tree-structured directories and the use of paths occurs later in this chapter. Also refer to Chapter 6 for a detailed explanation of tree-structured directories.

Filenames and Extensions

A filename consists of one to eight characters that can be used as a primary description of the information in the file. The file extension is an optional one to three additional characters separated from the filename by a period; it further defines the information contents of the file. Figure 5.14 illustrates the relationship of the filename and file extension.

Both uppercase and lowercase characters can be used in filenames and extensions; DOS does not distinguish between the two. Depending on the version of DOS used, certain characters may not be usable in filenames and extensions. Table 5.4 lists the

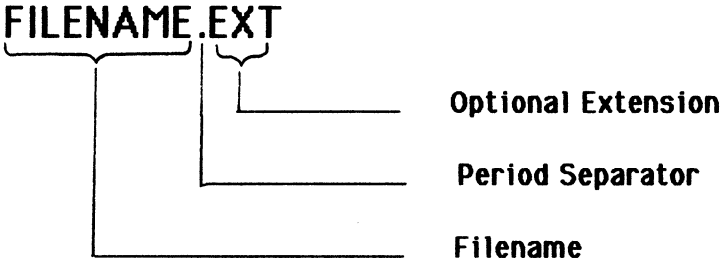


Figure 5.14
Filename and File Extension Relationship

Table 5.4
Invalid Filename and Extension Characters

Symbol	Name
"	Quotation
/	Slash
\	Backslash
[	Left bracket
]	Right bracket
¦	Broken vertical bar
=	Equal
:	Colon
;	Semicolon
<	Less than
>	Greater than
.	Period
,	Comma
	Space

invalid characters under either PC-DOS Version 3.3 or 4.0; they represent the majority of characters that cannot be used in filenames and extensions for all versions of DOS.

In assigning filenames and extensions, you should avoid certain names. These names are used by the operating system to reference specific components in the computer. Table 5.5 lists the DOS reserved names that cannot be used as filenames and the computer component or device they reference. Table 5.6 gives a list of reserved extensions and their meaning or DOS usage. Use these extensions *only* when you are creating one of the indicated file types or manipulating a previously created file with a DOS command. Otherwise, substantial confusion about the type of file can result, or the file may not operate correctly.

File-Naming Conventions

A key to maximizing the use of file reference commands is to establish and use consistent naming conventions. Although any group of legal characters can be used in developing

Table 5.5
DOS Reserved Names

Reserved Name	Device
CLOCK$	System clock device driver (DOS 4.0)
CON:	Console keyboard/screen
AUX: or COM1:	First serial communications port
COM2:	Second serial communications port
COM3:	Third serial communications port
COM4:	Fourth serial communications port
LPT1: or PRN:	First parallel printer port
LPT2:	Second parallel printer port
LPT3:	Third parallel printer port
NUL:	Nonexistent device for use in application program testing

Table 5.6
Reserved Extensions

Extension	Meaning
.BAK	Backup file
.BAT	DOS batch file
.CHK	Assigned to files recovered by CHKDSK
.COM	Program file directly executable by DOS
.EXE	Program file directly executable by DOS
.MAP	Default extension for list file created by DOS linker program
.OVL	Extension used by DOS for overlay files
.REC	Extension used by DOS for RECOVERed files
.SYS	Extension used by DOS for files containing system configuration and device drivers
.$$$	Extension used by DOS for temporary files

filenames and extensions, naming conventions generate standards that both serve to boost productivity and to eliminate vagueness that can result in other users spending minutes or hours searching for a particular file.

The development of specific naming conventions depends on your or your organization's requirements. As an example, a file containing accounts payable information for 1988 might be named ACTPAY88.DAT, with the extension .DAT used to indicate that the file is a data file.

Naming conventions for filenames can be easily developed to fit a particular application. In comparison, a large number of file extensions have been predefined and are accepted as de facto standards in addition to those extensions in Table 5.6. Table 5.7 lists, in alphabetical order, de facto file extension standards.

Table 5.7
De facto File Extension Usage

Extension	File Type
.ASM	Assembly language program in source code
.BAK	Backup file
.BAS	BASIC program
.BAT	Batch file containing DOS commands
.BIN	Binary file
.CHK	File recovered by CHKDSK
.COB	COBOL language program in source code
.COM	Command or program directly executable by DOS
.DAT	Data file
.DOC	Document file usually created by word processor
.EXE	Executable relocatable program
.FOR	FORTRAN program in source code
.LIB	Library program
.MAC	Macro for assembly language program
.MAP	Link program listing
.OBJ	Machine language (object) version of compiled program
.OVL	Application program overlay file
.OVR	Compiler program overlay file
.PAS	Pascal language program in source code
.PIC	Screen (picture) display image
.REC	Recovered file
.SYS	System configuration file
.TMP	Temporary file
.TXT	Text file
.$$$	Temporary file

Device Names

You can use names assigned to specific physical devices in DOS to direct the results of commands to those devices. Consider the names assigned to devices to be reserved names. As such, they cannot be used as diskette or fixed disk filenames, because the operating system assumes that they are assigned to specific physical devices.

Table 5.5 lists the names assigned by DOS to physical devices. If the console is used as an input device, the F6 key followed by Enter or Ctrl+Break can be used to terminate use of the console. Either action generates an end-of-file mark or character indication to DOS, which then terminates the operation. Concerning the NUL reserved name, the physical device assigned to this name is a dummy or nonexistent device that is used for test purposes. If you use this reserved name for testing as an input device, an immediate end-of-file character is generated. If this reserved name is used for testing as an output device, the write operations are simulated; however, no data is actually transferred. The NUL device can be very valuable in testing operations of batch files (collections of DOS commands). As an example, using the NUL reserved name instead of a printer name you can test the logical structure of the operation of the commands in the file without having to print data. Batch files are covered in detail in Chapter 8.

Note that the colons that follow reserved names are optional. In addition, DOS ignores any drive parameter or filename extension erroneously entered with a reserved name.

Paths

The introduction of large storage capacity in the form of fixed disks with the PC XT was accompanied by a major revision to DOS: Version 2.0. Although several major and minor revisions to DOS have occurred since Version 2.0, the hierarchical file structure of that version remains the means for effectively working with a large number of files.

Under the hierarchical file structure of DOS Versions 2.0 and later, a root directory is automatically created on each diskette and fixed disk when the media is formatted. The root directory of a specific disk drive is indicated by a drive designator followed by a backslash. For example, A:\ is the root directory of a diskette in drive A, whereas C:\ is the root directory of the fixed disk assigned to drive C.

A mixture of files and other directories can be placed under the root directory, with the other directories commonly called *subdirectories,* because they are nested under the root directory. Figure 5.15 illustrates an example of a directory structure that, although appropriate for a fixed disk, could also be used on a diskette.

In examining Figure 5.15, note that two subdirectories have been established directly under the root directory—DOS and WP. Under the DOS subdirectory you might locate a majority of DOS utility programs to facilitate their use when required, which is what the SELECT program does under DOS 4.0. Under the WP subdirectory, a word processing program named EW.COM and word processing files are stored. Assuming that two operators use the word processor, it might be appropriate to separate their data files. To accomplish this, separate subdirectories could be established, with each person storing data files in his or her subdirectory. This is illustrated by the GIL and BEV subdirectories in Figure 5.15.

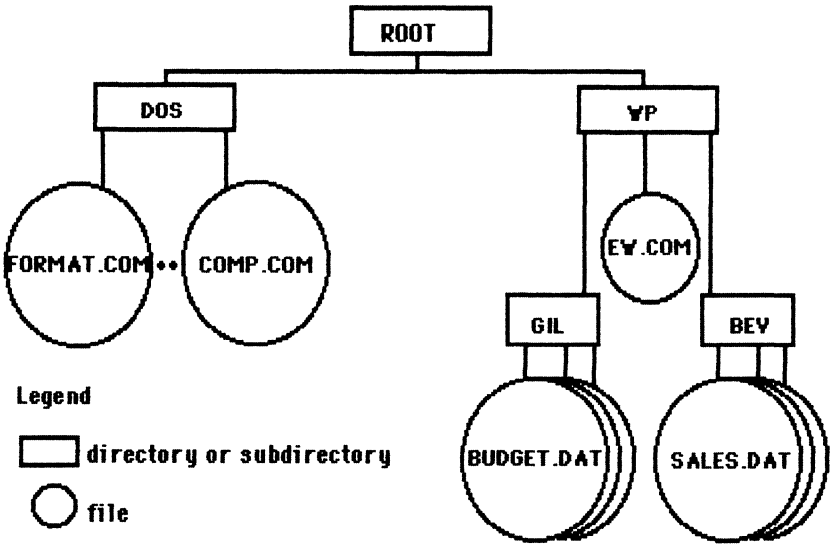


Figure 5.15
Sample Hierarchical Directory Structure

The use of a path to specify the route through the hierarchical file structure provides the mechanism for locating files and subdirectories. This in turn results in the requirement to add a pathname to the file specification.

In essence, the path is the route through directory names to create or access a file or subdirectory. The path consists of one or more directory names, each preceded by a backslash (\). If the path begins with a backslash, DOS starts its search from the root directory; otherwise, the search commences at the current directory. When a filename is included in the path it must be separated from the last subdirectory name by a backslash.

In examining Figure 5.15, the path to the file BUDGET.DAT from the route directory can be entered as

\WP\GIL\BUDGET.DAT

If the file is located on the fixed disk (drive C) and the default drive is drive A, the complete file specification required to access BUDGET.DAT becomes

C:\WP\GIL\BUDGET.DAT

Figure 5.16 illustrates the complete file specification broken down by its components.

In the preceding example, the drive identifier is optional if drive C is the default drive. Similarly, the path is optional if the user previously entered a DOS command to establish the subdirectory GIL as the current directory. Chapter 6 reviews DOS commands used for creating and navigating directories.

The format of a complete file specification follows; the items enclosed in brackets are optional.

[*d:*][*pathname*][*filename*[*.ext*]]

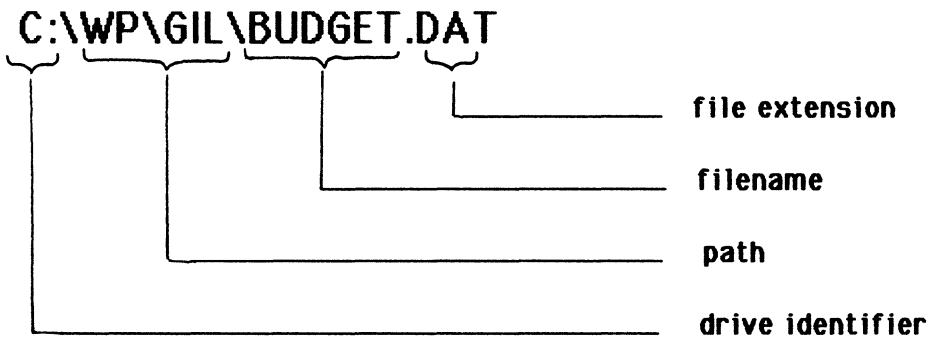


Figure 5.16
File Specification Components

Table 5.8
Global Character Usage Examples

Global Character Example	Meaning
.	All filenames
*.XYZ	All filenames with the extension .XYZ
XYZ.*	Any file named XYZ regardless of its extension
SAMPLE*.*	Any file whose name begins with SAMPLE, regardless of its extension
???PAY87.*	Any file with PAY87 in positions 4 through 8 of the filename, regardless of its extension

Global File Symbols

The question mark (?) and asterisk (*) are two special symbols you can use in filenames and extensions to mean "any character." Due to the comparable usage of these symbols to a special type of card in a card game, they are also commonly known as *wildcards.*

The question mark (?) character matches any single character in a filename or extension. This means that any one character can be in the position of the ?. For example, entering the directory (DIR) command

DIR ACTPAY??.???

causes any file in the directory that has ACTPAY in positions one through six of the filename to be displayed, regardless of its extension.

The asterisk (*) matches all characters from the asterisk's position to the end of the filename or extension. Thus, you can enter **DIR ACT*.*** to display all files in the directory that have ACT in positions one through three of their filenames, regardless of their extensions. Table 5.8 lists five examples of the use of global characters and their meaning.

DOS Commands

To use DOS effectively, you should have a firm understanding of the operation and usage of the commands included in the system. Although there are over 60 DOS commands you can use, focus initially on the small subset of those commands in this chapter. This subset of DOS commands are those most frequently used to perform a core of day-to-day computer operations. Knowledge of the operation and usage of these commands will enable you to perform such basic functions as setting and resetting the computer's date and time, formatting a disk, copying files from one storage medium to another, and obtaining a directory listing of files on a storage medium.

After reviewing the operation of these commands, this chapter concludes by examining the uses of the DOS Shell feature of DOS 4.0.

Internal Versus External Commands

Table 5.9 lists 25 commonly used DOS commands whose structure, operation, and usage will be covered in the remainder of this chapter. As indicated by the Type column in Table 5.9, DOS commands can be categorized as internal or external. The coding to process internal commands is contained in the command interpreter file, COMMAND.COM, which is read into memory when the operating system is initialized. Thereafter, when you enter the name of an internal command the command interpreter file—which is now memory resident—executes the appropriate coding to process the command. Thus, internal commands are also referred to as *memory-resident* commands.

The coding for the processing of external commands is contained in program files that are fixed disk- or diskette-resident, depending on where your operating system resides. When you enter an external command, COMMAND.COM recognizes that the coding resides in a file and uses the default drive and current subdirectory or a specified drive and path included with the command entry to locate the file. Once located, the contents of the file are loaded into memory and executed.

External command files can be easily recognized in a directory listing, because their filename extension is either .BAT, .COM, or .EXE. One example of a frequently used external command is the FORMAT command. The coding that processes and executes this command resides on the FORMAT.COM file on your DOS diskette. When you enter the command FORMAT, DOS searches the default drive or a specified drive for that file, loads the file once it is located, and then executes its contents. When you enter an external command you can omit its filename extension, because it is optional. The following examples illustrate the use of the FORMAT command to initialize diskette and fixed disk media for data recording. Each of the following examples shows the default drive using the prompt displayed by DOS 3.3. If you are using DOS 4.0, the prompt displayed when drive A is the default drive is `A:\>`, whereas the prompt displayed when drive C is the default drive is `C:\DOS>`.

FORMAT Command	*Result*
A>C:FORMAT	Formats diskette in drive A, which is the default drive, using the FORMAT.COM file located on drive C.

Table 5.9
Commonly Used DOS Commands

Command	Type	Activity Performed
ASSIGN	E	Routes disk I/O requests from one drive to another.
ATTRIB	E	Sets and resets the attribute byte and archive bit of a file or displays the status of the attribute byte and archive bit.
BREAK	I	Instructs DOS to check for a control break whenever a program requests DOS to perform an I/O operation.
CHKDSK	E	Checks the disk and displays a status report about its contents and your computer's memory.
CLS	I	Clears the display screen.
COMP	E	Compares the contents of two files.
COPY	I	Copies a specified file or set of files to the same or another disk.
DATE	I	Displays or stores a date in your computer.
DEL	I	Deletes a specified file or set of files (same as ERASE).
DIR	I	Displays the files stored on a disk that match your specifications.
DISKCOMP	E	Compares the contents of one diskette to another.
DISKCOPY	E	Copies the contents of one diskette onto another.
ERASE	I	Deletes a specified file or set of files (same as DEL).
FORMAT	E	Prepares a diskette or disk for use and optionally copies the operating system files to it.
GRAFTABL	E	Loads a table into memory that defines ASCII characters 128 through 255.
GRAPHICS	E	Permits the contents of the color graphics video display mode to be printed.
LABEL	E	Creates, changes, or deletes a volume label on a disk.
PROMPT	I	Sets a new DOS prompt.
RENAME	I	Changes the name of a file or set of files.
SYS	E	Transfers the operating system files.
TIME	I	Displays or stores the time in your computer.
TYPE	I	Displays the contents of a file on the screen.
VER	I	Displays the version of DOS you are using.
VERIFY	I	Verifies the data written onto a disk was correctly recorded.
VOL	I	Displays the disk volume label of a specified disk.

A>FORMAT B: Formats the diskette in drive B using the FORMAT.COM file located on drive A.

C>FORMAT A: Formats the diskette in drive A using the FORMAT.COM file located on drive C.

C>**FORMAT**	Formats the fixed disk using the FORMAT.COM file located on drive C.

These four examples of the FORMAT command limited the use of command parameters to a drive specifier. Later, this chapter examines the optional parameters that can be included in this command.

The fourth example illustrates how easy it is to inadvertently format your fixed disk. In that example, no drive specifier was included in the command line, causing the command to be executed on the default drive—the fixed disk.

Due to the potential effect of destroying tens to hundreds of millions of bytes of data by an inadvertent format of the fixed disk, DOS displays a warning message when you attempt to format a fixed disk that was previously formatted. Earlier versions of DOS did not include a warning message, and many persons inadvertently reformatted their fixed disk.

DATE (Internal)

The DATE command displays the current date known to your computer system and if you wish, changes that date. The format of this command is

DATE { *mm-dd-yy* | *dd-mm-yy* | *yy-mm-dd* }

If you enter the command **DATE** by itself, DOS displays the current date and prompts you to enter a new date:

```
A>DATE

Current date is Fri 5-9-1989

Enter new date (mm-dd-yy):
```

Once the current date is displayed, you can enter a new date or press the Enter key to leave the current date unchanged. The actual format in which the current date is displayed, as well as the format you will use to enter a new date, depends on the country code used when you installed DOS. In North America, the month-day-year format (*mm-dd-yy*) is commonly used, with the other two DATE formats used primarily in European countries.

When you enter the DATE command with parameters or respond to the `Enter new date` prompt, the following constraints must be adhered to or DOS generates an `Invalid date` message.

m must be 1 or 2 digits from 1 to 12
d must be 1 or 2 digits from 1 to 31
y must be 2 digits from 80 to 99 or 4 digits from 1980 to 1999

You can separate the parts of the date using a hyphen (-), slash (/), or period (.). The following examples illustrate the use of this command.

```
A>DATE
Current date is Thu 6-16-1988
Enter new date (mm-dd-yy):
```

```
A>DATE 6-16-89
```

```
A>DATE
Current date is Fri 6-16-1989
Enter new date (mm-dd-yy): 6-16-88
```

```
A>DATE

Current date is Thu 6-16-1988
Enter new date (mm-dd-yy):
```

If a calendar is not available, the DATE command provides an equivalent mechanism to determine the day of the week for a particular date. As an example of this, say you enter the DATE command with the parameters 5-15-98. DOS informs you that that date is a Friday. This information displays when you enter the DATE command a second time, as indicated here.

```
C>DATE 5-15-98
```

```
C>DATE

Current date is Fri 5-15-1998
Enter new date (mm-dd-yy):
```

If you use the DATE command as a calendar, be sure to enter the current date in response to the `Enter new date` prompt. This action is required to reset your computer's date and is important to remember, because the system date is recorded in the directory whenever you create or modify a file.

TIME (Internal)

The TIME command displays or changes the time known to your computer system. The format of this command is

```
TIME [hh[:mm[:ss[.xx]]]]
```

where hh is 1 or 2 digits from 0 to 23 that represents hours, mm is 1 or 2 digits from 0 to 59 that represents minutes, ss is 1 or 2 digits from 0 to 59 that represents seconds, and xx is 1 or 2 digits from 0 to 99 that represents hundredths of a second.

Because the format or syntax of the TIME command may appear confusing, take a moment to review it. Because entries in brackets are optional, the TIME command can be entered without any parameters. You can also enter the command with just an hour; with an hour and minute; with an hour, minute and second; or with an hour, minute, second, and hundredth of a second.

The hour parameter is expressed in military time, with 1 p.m. entered as 13, 2 p.m. as 14, and so on. Whereas a colon is used to separate hours from minutes and minutes from seconds, a period must be used to separate seconds from hundredths of a second.

If you enter the command without any parameters, you see the display of the current time and a prompt message to enter the new time:

```
A>TIME

Current time is 9:59:02.91
Enter new time:
```

To leave the time as currently displayed, press the **Enter** key without entering any values. If you only enter one parameter, such as a new hour, the remaining parameters are initialized to zero. This is partially illustrated by the following example.

```
A>TIME

Current time is 9:57:37.06


Enter new time: 11

A>TIME

Current time is 11:00:02.65
Enter new time:
```

In the preceding example, an 11 was entered for the new time. When the TIME command was entered a second time, the minute field was zero, but 2 and 65 hundredths of a second transpired between entering the new time and displaying the current time.

DOS Shell Date and Time Setting

Using the DOS Shell to set the date and time, as with many other Shell functions, you need not remember command formats. This is because the DOS Shell simplifies the execution of many commands by displaying pop-up boxes that illustrate the format of the data to be entered. In addition, a help facility can be invoked by pressing the F1 key. It provides further information concerning the activity you are performing.

When the Start Programs screen in the Shell is initially displayed, as shown in Figure 5.10, the Command Prompt item in the Main Group is highlighted. Using the Down arrow key, you can move the selection cursor over the DOS Utilities entry. Pressing the Enter key selects this item in the group area, resulting in the DOS Utilities screen display. This screen is illustrated in Figure 5.17. From this screen you can select from six DOS utilities, including one to set the date and time.

When the DOS Utilities screen is displayed, the selection cursor is positioned over the `Set Date and Time` entry. Pressing Enter to select this entry results in the display of a set of pop-up boxes, the first of which is illustrated in Figure 5.18. When each pop-up box is displayed, a cursor is positioned at the first data entry position. As indicated by the row at the bottom of the pop-up box, you can press the Enter key to have DOS accept the data you entered, press the Esc key to cancel the operation, or press the F1 key for help. Pressing F1 displays a pop-up box.

Once you enter the date, a second pop-up box for entering the time is displayed. Unlike the command prompt mode of DOS where the time can be set to hundredths

Figure 5.17
DOS 4.0 Utilities

of a second, under the DOS Shell the time is set to the nearest minute. Thus, you may have to wait up to 59 seconds to enter a precise time when you use the DOS Shell.

After you complete the date and time pop-up box items, you can return to the Start Programs Main Group screen previously illustrated in Figure 5.10 if you select an appropriate entry from the `Exit` item in the action bar. To make the selection, press **F10** to select the action bar, which highlights the `Program` item. Once this is accomplished, you can either type **X**, which is underlined in the `Exit` item, or you can press the Right arrow key to reposition the selection cursor over `Exit` and press the **Enter** key. This displays an Exit pull-down box. This pull-down box contains two entries—`Exit Shell` and `Resume Start Programs`—with the selection cursor positioned over the top item, Exit Shell. Note the first characters in the choices; underlined characters show which letters you can type. Alternatively, you can move the selection cursor over the Resume Start Programs entry and press the Enter key to return to the screen illustrated in Figure 5.10.

FORMAT (External)

As discussed earlier in this chapter, the FORMAT command initializes a disk for recording data. During the disk initialization process, the FORMAT command analyzes the media for defective tracks and creates a directory and File Allocation Table (FAT). The directory is an area on the disk reserved to hold information about a file, subdi-

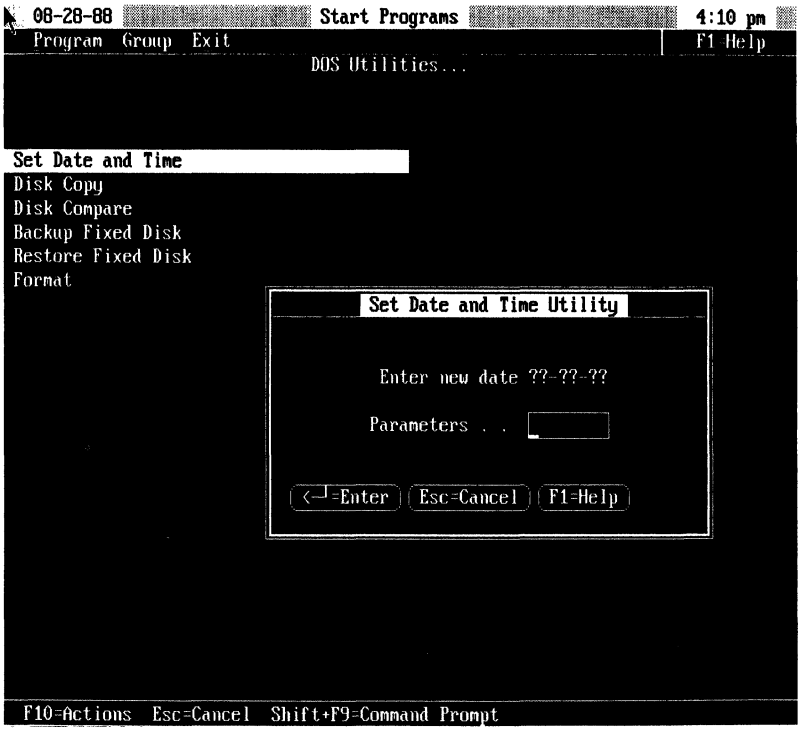


Figure 5.18
Setting the Date and Time Under DOS 4.0

rectory, or a volume label. The FAT is an area on the disk that serves as a pointer to the location where individual files reside on the disk.

Because formatting destroys all previously recorded data on the diskette or fixed disk, be very careful to ensure that the medium to be formatted is correct. This is especially true for the fixed disk, whose inadvertent formatting can destroy tens to hundreds of millions of bytes of data.

The syntax of the FORMAT command under DOS 3.3 is

```
[d:][path]FORMAT
d:[/S][/1][/8][/V][/B][/4][/N:xx][/T:yy]
```

Under DOS 4.0, the optional parameter /V was changed to /V:label, and a new optional parameter, /F:size, was added to the syntax of the command.

The optional drive identifier and path preceding the command keyword are only required if the file FORMAT.COM is at a location other than the default drive or if it's located in a directory other than the current directory. Thus, the first command line (for DOS 3.3) and the second command line (for DOS 4.0)

```
A>C:FORMAT B:
```

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

each tell DOS to retrieve the file FORMAT.COM from the current directory of drive C to format a diskette in drive B. Following the drive letter of the device to be formatted, you can enter a subset of eight (DOS 3.3) or nine (DOS 4.0) optional parameters. The actual parameters you can specify depend on the medium you will format (diskette or fixed disk), the type of diskette drive in which a diskette is installed, and previously entered command line parameters, because some parameters are mutually exclusive of one another.

By specifying the /S option, you copy the three operating system files—IBMBIO.COM, IBMDOS.COM, and COMMAND.COM—to the formatted disk at the conclusion of the formatting operation. Figure 5.19 illustrates the use of the /S option in the FORMAT command. In this example, DOS resides on a diskette in drive A and the format operation is performed on a diskette in drive B.

If you are using DOS 4.0 and have a fixed disk, the INSTALL program you used to set up your system created an AUTOEXEC.BAT file with a PATH statement that contains the route to the subdirectory DOS on drive C. This means that DOS automatically searches the C:\DOS disk/directory location for the file FORMAT.COM if it does not reside on the diskette in drive A. Once the format operation is invoked, under DOS 4.0 a `percent of disk formatted` message displays as the disk is being formatted and is erased from the screen when the operation is complete.

The message `System transferred` indicates that three system files were copied to the newly formatted disk.

Using the /S parameter in the FORMAT command makes the disk self-booting. This means you can place the resulting disk in drive A and use it to bring up your computer. The system files transferred to that disk prompt you to enter the date and time and display the copyright notice prior to displaying the prompt `A>` or `A:\>` under DOS 3.3 and DOS 4.0, respectively. After you format a diskette using the /S parameter, you can copy application programs to that diskette, so your application diskette becomes self-booting. This can be very handy for use on a PS/2 computer that has only one diskette drive for distributing the application to users with that hardware configuration. This is because a self-booting diskette eliminates the necessity to first load DOS and then to remove the DOS diskette to use an application diskette.

Figure 5.19 Using the FORMAT /S Option

```
A:\>FORMAT B:/S
Insert new diskette for drive B:
and press ENTER when ready...

Format complete
System transferred

Volume label (11 characters, ENTER for none)? FINANCE

   1457664 bytes total disk space
    107520 bytes used by system
   1350144 bytes available on disk

       512 bytes in each allocation unit
      2637 allocation units available on disk

Volume Serial Number is 2476-11FA

Format another (Y/N)?
```

Another difference between DOS 3.3 and DOS 4.0 concerns volume label prompting. Under DOS 3.3, unless you enter the /V parameter you are not prompted to enter a volume label. Under DOS 4.0, you are automatically prompted to enter a volume label unless you use the /V:label option in the command line to specify a label. Doing so results in DOS accepting the label and bypassing the display of a prompt asking you to enter a label.

The volume label uniquely identifies each disk and can be checked using the DIR command. Both the /S and /V parameters can be included in a FORMAT command, resulting in the transfer of system files to the disk, as well as the placement of a volume label on the media.

In general, as you assign volume labels to diskettes, be as explicit as possible about the contents of the media. Otherwise, simply assigning a volume label, such as FINANCE, to several diskettes can necessitate a substantial effort later to locate a diskette containing July 1989 financial data files.

The remaining format parameters are related to the different types of diskettes supported by the original IBM PC series and the PS/2 family of personal computers. The /1 parameter formats a 5¼-inch diskette for single-sided use. The original IBM PC was introduced with single-sided disks in 1981. The /1 parameter enables you to format a diskette in a 5¼-inch drive attached to your PS/2, so that data from your 3½-inch diskettes or fixed disk can be copied to 5¼-inch media that is compatible with computers using single-sided diskette drives.

The /8 parameter formats a diskette for recording data 8 sectors per track. This was the original recording method used by DOS Versions 1.0 and 1.1. Later, when DOS Version 2.0 was introduced, the FORMAT command would default to 9 sectors per track for conventional 5¼-inch drives and 15 sectors per track on high-capacity diskettes installed in a 1.2M byte high-capacity diskette drive.

The /B parameter formats a diskette for 8 sectors per track as well as reserve space for two hidden system files, IBMBIO.COM and IBMDOS.COM. This parameter should be used with the FORMAT command to create a diskette on which any version of DOS can be placed by using that version's SYS command.

The /4 parameter formats a conventional double-sided diskette in a 1.2M byte high-capacity drive. Because only conventional 5¼-inch diskette drives are originally marketed for use with members of the PS/2 series, it is doubtful whether you will ever use this parameter when you use those drives.

The /N:xx parameter specifies the number of sectors per track to format, with xx used to represent the number of sectors. The /T:yy parameter specifies the number of tracks to format. Both parameters are used in the FORMAT statement when you want to format a diskette to a capacity less than the maximum supported by the diskette drive. As an example of the use of these parameters, assume you have a PS/2 Model 50 and wish to share data files with a PS/2 Model 30 computer. The diskette drives used in the Model 50 have a storage capacity of 1.44M bytes, whereas the drives used in the Model 30 have a storage capacity of 720K bytes. If you use the FORMAT command without any parameters, the Model 50 formats a diskette with a capacity to store 1.44M bytes of data. This diskette cannot be used in the Model 30, because the diskette drives in that computer are limited to a storage capacity of 720K bytes. To ensure compatibility

with the Model 30's diskette drives, you enter the parameters /N:9/T:80 in the FORMAT command to format diskettes in a disk drive on the Model 50 for use on a Model 30.

The /F:size parameter was added to the FORMAT statement when DOS 4.0 was released. This parameter was included to provide a mechanism for computer users to easily specify a format for a medium with a storage capacity less than the maximum capacity of a diskette drive. When the /F:size parameter is used in the FORMAT statement, you can specify the storage capacity as a decimal number by itself or by adding several types of suffixes to the number to denote kilobyte or megabyte storage capacity.

Table 5.10 lists the permissible /F:size values based on diskette type. Note that you can only specify a lesser capacity disk format in a higher capacity drive and the specifications must be media type compatible. That is, you cannot specify a 360K byte disk format associated with a 5¼-inch diskette in a 3½-inch diskette drive, nor could you specify a 720K byte disk format associated with a 3½-inch diskette if you are using a high-capacity 1.2M byte 5¼-inch disk drive.

The data in Table 5.11 indicates the FORMAT parameters permitted based on the type of disk to be formatted.

DOS Shell FORMAT Process

Similar to setting the date and time, you can initiate the formatting process under the DOS 4.0 Shell by selecting the `DOS Utilities` option from the Start Programs Main Group screen, shown earlier in Figure 5.10. Once this is accomplished, the DOS Utilities

Table 5.10
/F:size Values Based on Diskette Type

Disk Type	Permissible Values					
160K bytes	160,	160K,	160K bytes			
180K bytes	180,	180K,	180K bytes			
320K bytes	320,	320K,	320K bytes			
360K bytes	360,	360K,	360K bytes			
720K bytes	720,	720K,	720K bytes			
1.2M bytes	1200,	1200K,	1200K bytes,	1.2,	1.2M,	1.2M bytes
1.44M bytes	1440,	1440K,	1440K bytes,	1.44,	1.44M,	1.44M bytes

Table 5.11
Format Parameter Usage

Disk Type	Parameters Allowed
160K byte/180K byte	/S, /V, /1, /8, /B, /4, /F:size
320K byte, 360K byte	/S, /V, /1, /8, /B, /4, /F:size
720K byte, 1.44M byte	/S, /V, /N, /T, /F:size
1.2M byte	/S, /V, /N, /T, /F:size
Fixed disk	/S, /V

screen illustrated in Figure 5.17 is displayed. Move the cursor selection bar with the **Up** or **Down arrow** key until it is positioned over Format. Then press **Enter** to display a pop-up box labeled `Format Utility` on the DOS Utilities screen, as illustrated in Figure 5.20. When this pop-up box is displayed, drive A is the default drive parameter in the Parameters box, with the cursor positioned under the letter `a`. You can accept that drive, enter a different drive, and enter any applicable FORMAT parameters at this time. Assuming you entered appropriate FORMAT parameters, pressing Enter invokes the FORMAT program, prompting you to insert a new diskette for drive A and to press the Enter key when you are ready.

After the format operation is completed, you are prompted to enter a volume label unless you entered the parameter /V:label in the Parameter box. Similar to the FORMAT operation previously illustrated in Figure 5.19, information concerning disk space and the volume serial number are displayed, followed by the prompt `Format another (Y/N)?`. Assuming you enter N, pressing any key in response to a `Press any key` prompt causes the DOS Utilities screen in the DOS Shell to be redisplayed.

Although the DOS Shell facilitates elementary formatting, you must remember the availability and syntax of optional parameters to make full use of this command. In fact, the Help pop-up box displayed in response to pressing the F1 key refers you to the IBM *Using DOS* book for information concerning the optional parameters you can use.

Figure 5.20
DOS 4.0 FORMAT Utility

CLS (Clear Screen) (Internal)

If you are working at the command prompt level and have entered a few of the examples covered in this chapter, your display screen may be cluttered with information. The CLS (clear screen) command erases previously displayed data with the exception of the DOS prompt, which is redisplayed. The format of this command is simply the command name:

```
CLS
```

One popular use of the CLS command is to clear the screen of previous activity before you print a copy of some screen operation. Thus, enter **CLS**, followed by the actions you want to print, then press **PrtSc** to generate a hardcopy of desired activity that is not cluttered with other information. There is no equivalent command under the DOS Shell, because operations are based on the selection of items and filling boxes in pop-up screens that are automatically cleared after selection.

BREAK (Internal)

Before you examine a series of commands that can be used to perform multiple operations, an examination of how to terminate DOS command mode operations is warranted. Normally, you can press the **Ctrl+Break** multikey combination to terminate a DOS command. Even under the DOS Shell, you can press this key combination to terminate the operation of an activity, such as formatting a diskette.

DOS normally checks for the Ctrl+Break multikey sequence when data is entered from the keyboard, displayed on the monitor, or printed. To increase the ability to break out of a program that produces few or no standard device operations—such as a computational intensive program—you can use the BREAK command. The format of this command is

```
BREAK [{ON }]
      [{OFF}]
```

When you use BREAK without a parameter, the current state of the command (ON or OFF) is displayed. When BREAK is set to ON, DOS continuously checks for Ctrl+Break. Otherwise, when BREAK is set to OFF (its default value), DOS only checks for Ctrl+Break during standard input/output operations. The following example illustrates checking the state of the BREAK command under DOS 3.3 and setting its default value of OFF to ON. Under DOS 4.0, the only change is in the DOS prompt, which is initially set to C:\> if your system has a fixed disk.

```
C>BREAK
BREAK is off

C>BREAK ON

C>BREAK

BREAK is on

C>
```

DIR (Directory) (Internal)

The DIR (Directory) command helps you determine the file and subdirectory contents of a storage medium. Using this command, you can obtain information about all directory entries or the entries for a single file or group of specified files. The format of this command is

```
DIR [d:][path][filename[.ext]][/P][/W]
```

The optional drive specifier (d:) indicates the drive for which a directory listing should be taken. If the drive identifier is omitted, the directory listing will be taken for the default drive's current directory. Thus,

A>**DIR** (DOS 3.3)

A:\>**DIR** (DOS 4.0)

generates a directory listing of the contents of a diskette in drive A. Modifying the command line to

A>**DIR B:** (DOS 3.3)

A:\>**DIR B:** (DOS 4.0)

lists the directories and filenames of a diskette in drive B.

The path option obtains a directory listing when a hierarchical or tree-structured storage medium is used. By incorporating a path to the DIR command, you can obtain a directory listing of specific subdirectories or a file or group of files located in a specific subdirectory. Examples of the use of path parameters in DIR commands, as well as a detailed examination of subdirectory related commands, are given in Chapter 6.

To illustrate the use of the DIR command and its optional parameters, examples here apply the command against a common diskette. In the example illustrated in Figure 5.21, a directory listing of a diskette in drive B was taken by entering the command DIR B:. In this example, *xxxxxxx*, which represents the number of bytes available for data storage, depends both on the size of currently stored data files and the storage capacity of the disk.

In the directory listing illustrated in Figure 5.21, the backslash (\) after the message Directory of B: indicates it is a directory of the root or top of a hierarchical directory

Figure 5.21
Directory Listing of Diskette in Drive B

```
A>DIR B:

 Volume in drive B is DATACOM
 Directory of B:\

COMMAND    COM    25307    3-17-87    12:00p
BLAST      EXE   158772    4-13-87     7:59a
INDEX      EXE    13596    4-13-87     7:56a
INSTALL    EXE    28252    4-13-87     7:56a
        4 File(s)        XXXXXXX bytes free
```

structure. Unless a path is specified, the DIR command lists the contents of the current directory that, in this example, is the root directory.

You can obtain information about a specific file or a group of files by entering a filename and the ? and * global characters in the filename and filename extension. Table 5.12 lists three examples of the use of global (wildcard) characters and filenames in a DIR statement and the operational result of each command line entry.

The example illustrated in Figure 5.22 shows the use of the asterisk wildcard character on the diskette in drive B. As indicated in Figure 5.21, this diskette contains four files. Note that because the diskette only contains one file whose extension is .COM, only one file is included in the new directory listing shown in Figure 5.22.

The /P parameter automatically pauses the display when the screen is filled. This parameter is valuable when the directory you wish to list is extensive and entries would otherwise rapidly scroll off the screen. When you use /P, the following prompt is displayed:

```
Strike a key when ready...
```

At this time, you can press any key to continue with the directory listing.

The /W parameter obtains a "wide" display of the directory. When this parameter is used in a DIR command, each line in the directory contains up to five filenames. The example illustrated in Figure 5.23 shows the use of the /W parameter. Note that while the /W parameter displays up to five filenames on a line, the resulting directory listing excludes information concerning the size of the file and the day and date it was created or last modified.

DOS Shell File System Directory Operations

The DOS Shell File System performs a variety of file and directory operations, including display of different types of directory listings. Because you were just introduced to the

Table 5.12
DIR Command Examples

DIR Command	Operational Result
DIR A:*.*	Produces a directory listing of all files located on the diskette in drive A.
DIR B:*.EXE	Produces a directory listing of all files on the diskette in drive B whose extensions are .EXE.
DIR PAY*.*	Produces a directory listing of all files located on the diskette in the default drive whose filenames start with PAY, regardless of the files' extensions.

Figure 5.22
Using a Global Character in a DIR Command

```
A>DIR B:*.COM

 Volume in drive B is DATACOM
 Directory of B:\

COMMAND    COM    25307    3-17-87    12:00p
          1 File(s)          XXXXXXX bytes free
```

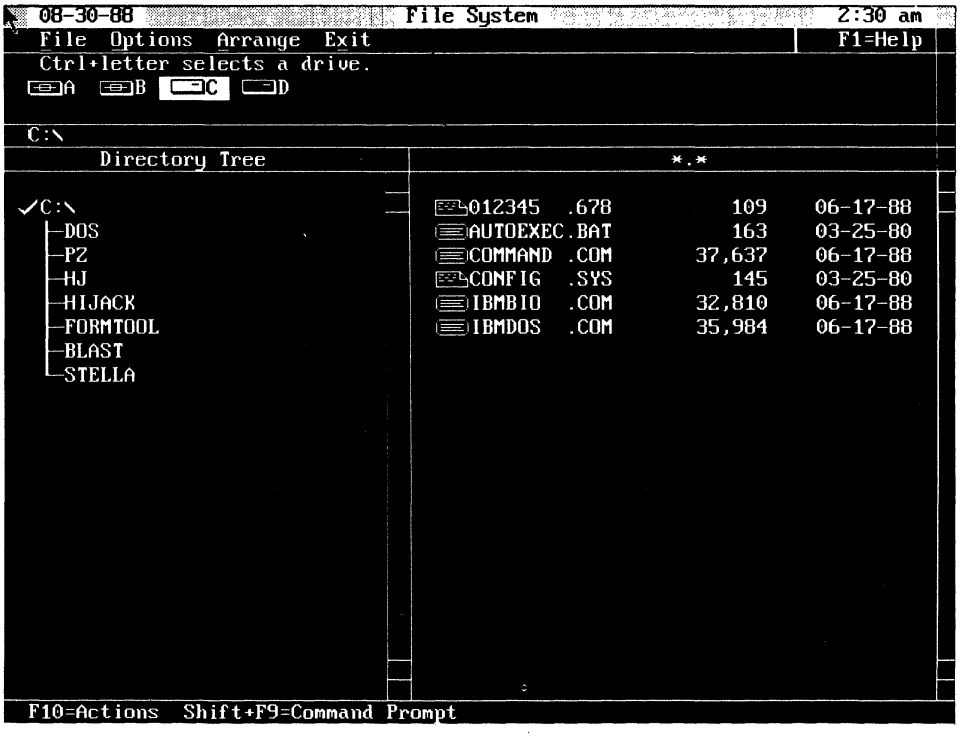
Figure 5.23
Using the /W Parameter in the FORMAT Command

```
A>DIR B:/W

 Volume in drive B is DATACOM
 Directory of B:\

COMMAND    COM   BLAST    EXE       INDEX      EXE    INSTALL    EXE
           4 File(s)        XXXXXXX bytes free
```

Figure 5.24
DOS 4.0 File System Screen



use of the DIR command, you can compare the use of the DOS Shell File System to perform equivalent operations.

The File System can be selected from the Start Programs Main Group previously illustrated in Figure 5.10 by moving the selection cursor over `File System` and pressing **Enter**. Figure 5.24 illustrates the File System screen.

After you select the File System, a directory listing of the default drive and directory is automatically performed, with the results displayed. In examining Figure 5.24, note in the upper left corner that the icon for drive C is highlighted; this means that drive C is the default drive. You can select another drive by pressing **Ctrl** and the drive designator letter. The row with the entry `C:\` under the drive identifier row displays the current path. In this example, it denotes the current location at the root directory on drive C.

The Directory Tree window along the left third of the screen displays the directory structure under the root directory of drive C. The right window displays the filenames that correspond to the selected directory. In this example, the check mark to the left of C:\ under Directory Tree indicates that the filenames in the root directory are to be displayed. The symbol *.* to the right of Directory Tree identifies the type of files to be displayed. When you invoke the File System, the default of *.* is used, which represents the global characters for all files and all extensions. Thus, the right window in Figure 5.24 displays all files in the root directory regardless of their extension. If you use the Up or Down arrow keys to select another directory, the filenames in this window automatically change. This is illustrated in Figure 5.25, where the subdirectory FORMTOOL was selected in the Directory Tree window, displaying all files in that directory regardless of their extension.

Press the **F10** key to move the selection cursor to the action bar. From there you can select any item in the action bar by entering the letter underlined in an item or using the **Left** or **Right arrow** key to position the selection cursor over the item. After an action bar item is selected, you can press the **Down arrow** key to display the pull-down menu associated with the item. Once a pull-down menu is displayed, you can use the Left or Right arrow key to access pull-down menus associated with the other items in the action bar.

The File pull-down menu options perform such file-related operations as renaming, copying, and viewing files, as well as the ability to create a directory. A later section

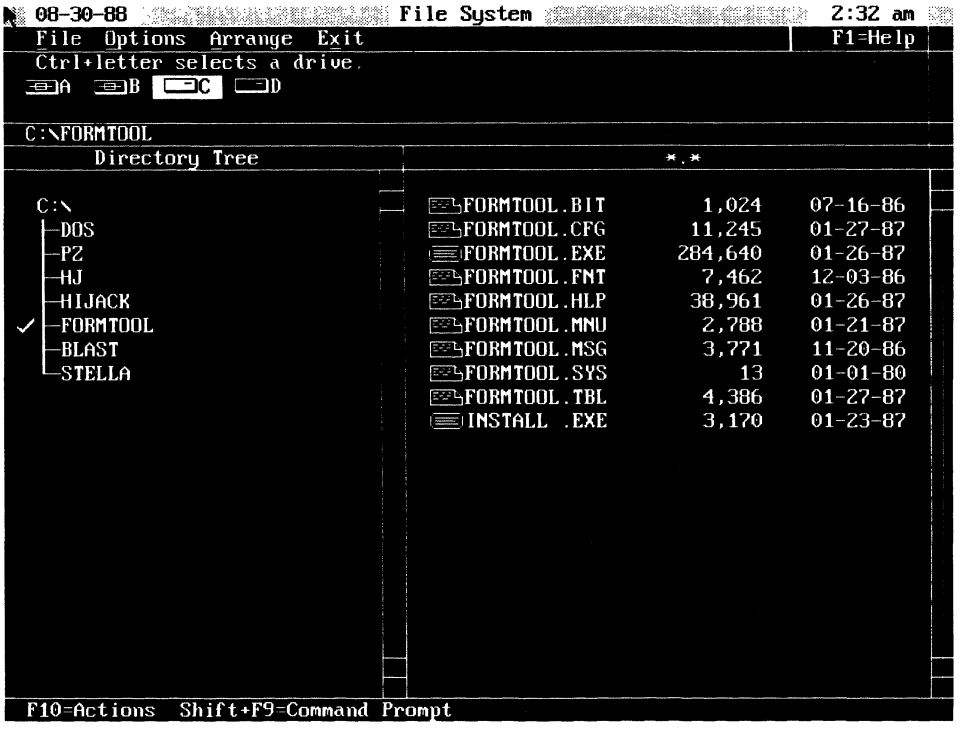


Figure 5.25
Changed Directory Under DOS 4.0 File System

of the chapter examines the use of portions of this pull-down menu; here the equivalent command prompt operations are described.

Select the Options item in the action bar and press **Down arrow** to display the Options pull-down menu shown in Figure 5.26. Selecting the Display options item generates a Display Options pop-up box over the other File System windows on your screen, as illustrated in Figure 5.27. This pop-up box is the key to obtaining information about a specific file; set of files with a common name, common extension, portion of a common name or extension; or all files in the directory. In addition, this pop-up box offers the capability to sort files in a directory five different ways. As indicated in Figure 5.27, initially the cursor is positioned at the first entry position, the Name box. If you accept the default entry *.*, all files in the currently checked subdirectory are displayed. Note that the button to the left of Name in the Sort by: column is highlighted. This is the default method of sorting directory entries. You can press the **Tab** key to move the selection cursor to highlight the Name entry in this column. Then use **Down arrow** to select a different sorting method, including sorting by Extension, Date, Size, or Disk order.

Although the DOS Shell File System facilitates the display of files better than by the use of command mode operations that have no sorting capability, to effectively use this Display Options box you must be familiar with the use of global (wildcard) filename characters. Thus, entering *.COM in the Name box in Figure 5.27 displays all files whose extension is .COM that are located in the root directory of drive C. Similarly,

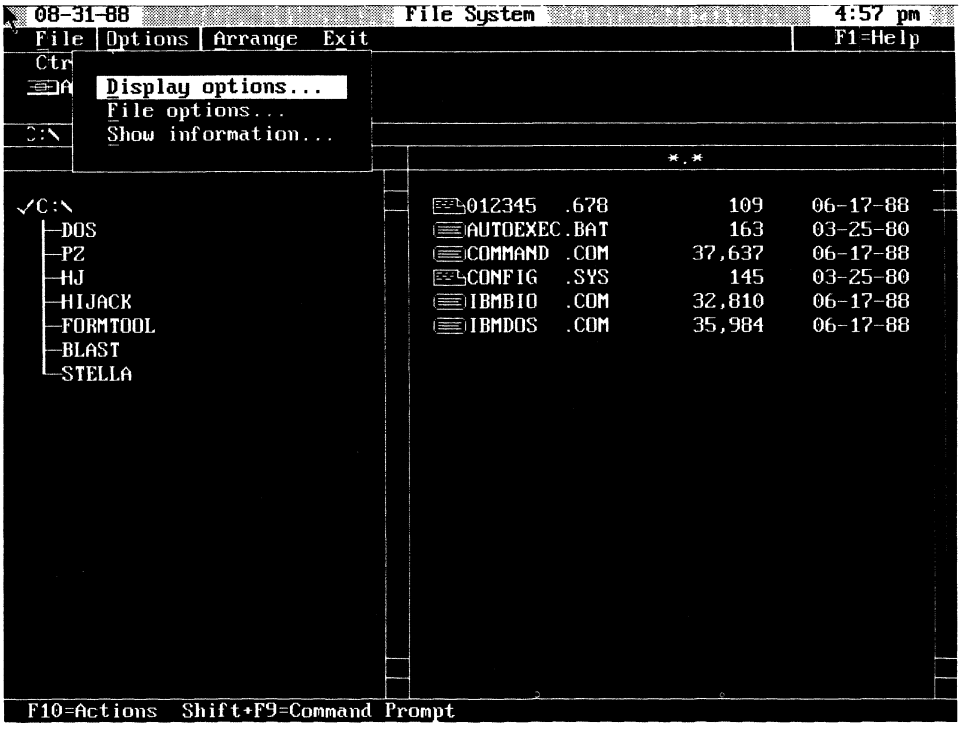


Figure 5.26
DOS 4.0 Display Options Pull-Down Menu

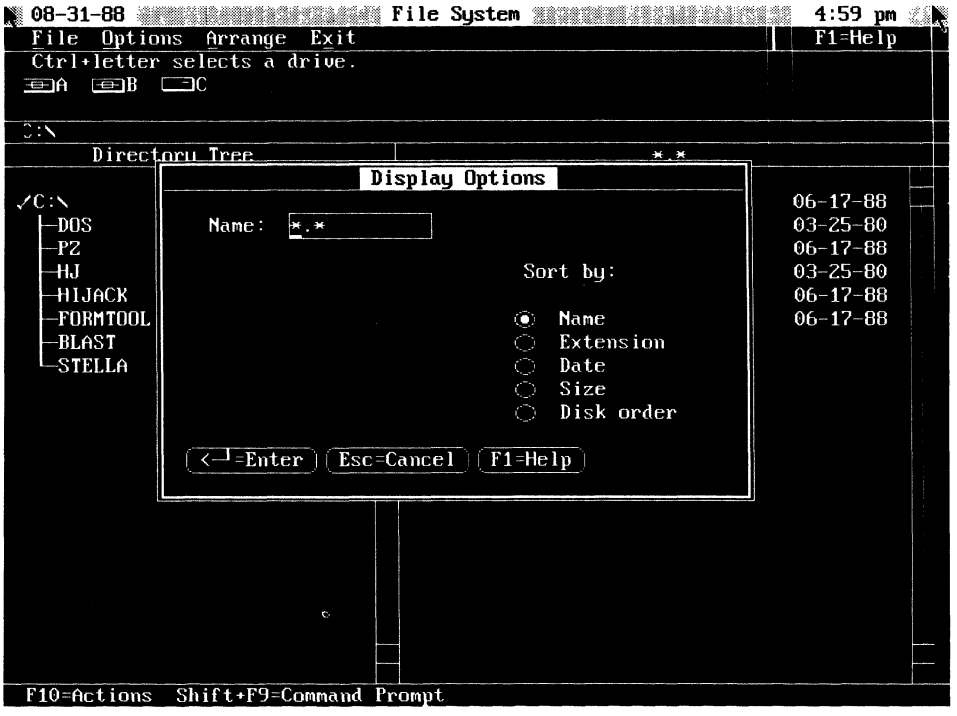


Figure 5.27
DOS 4.0 Display Options Pop-Up Box

entering C*.* displays all files whose names start with the letter C and are located in the root directory of drive C.

Select `File Options` in the Options pull-down menu to verify certain file related activities, as well as to display files in more than one directory at the same time. Figure 5.28 illustrates the `File Options` pop-up box. The default for the `Confirm on delete` and `Confirm on replace` boxes in the `File Options` pop-up box is checked (selected) here, meaning that a confirmation prompt is displayed before a file is deleted or moved or copied with a replace operation. The `Select across directories` entry enables you to select files in different directories. You can toggle the selection or deselection of any box item by pressing the **Up** or **Down arrow** key to move the selection cursor over the item. Then, press the **Space bar** to select or deselect the action; a check mark to the left of any item indicates it is selected.

The last item in the Options pull-down, `Show information`, displays information concerning a previously selected file or group of files or summarizes information concerning the current directory and the current drive. Summarized directory information includes the current directory name, total storage of files in the directory, and total number of files in the directory. Drive summary information includes the disk name, storage capacity denoted as `Size`, available storage capacity, labeled `Avail`, and the total number of files and directories on the disk. Thus, `Show information` can be viewed as an enhanced DIR command that provides information at the file, directory, and drive level.

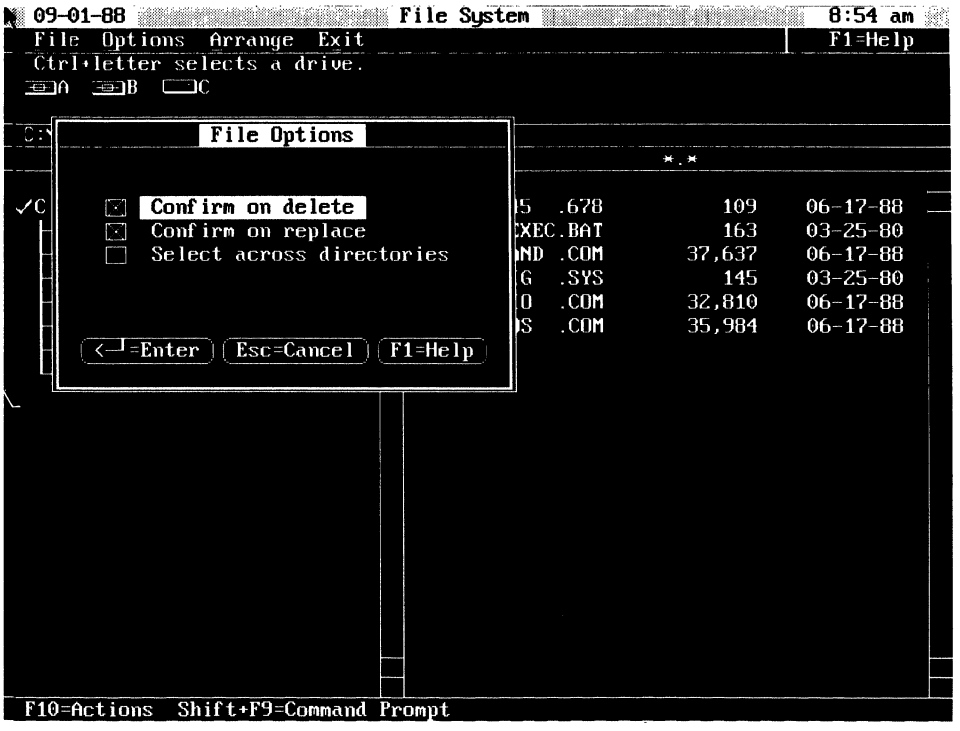


Figure 5.28
DOS 4.0 File Options

The `Arrange` item on the action bar provides a pull-down menu containing three entries, as illustrated in Figure 5.29. The first entry, `Single file list`, is the default method by which the DOS Shell File System displays the Directory Tree and associated filenames. That is, a single file list is used for display purposes.

Selecting `Multiple file list` entry essentially splits your screen horizontally into two wide windows with two "panes" each, as Figure 5.30 shows. Initially, the same Directory Tree and associated filenames are displayed in each list. Because the area for the top list is slightly smaller than the area used for the display of the second list and each area is less than a single list area, a short directory structure and small number of files may not be completely visible in either window. When this occurs, press **Tab** to move the selection cursor to the first entry in the Directory Tree or filename pane in either window. Then, you can use the **Up** and **Down arrow** keys to scroll through the contents of either pane.

One of the key benefits of the `Multiple file list` option is the capability it provides for viewing the contents of two separate directories at the same time. To display two directories, first use the **Tab** key to position the selection cursor at the top item in the Directory Tree area of either list. Then, position the selection cursor using **Up** or **Down arrow** so the selection cursor highlights the directory whose contents you wish to view. Finally, press **Enter** to display the contents of the directory in the right pane. Figure 5.31 represents the display of the contents of two directories at the same time. To recap, this display resulted from selecting the `Multiple file list` option in the Ar-

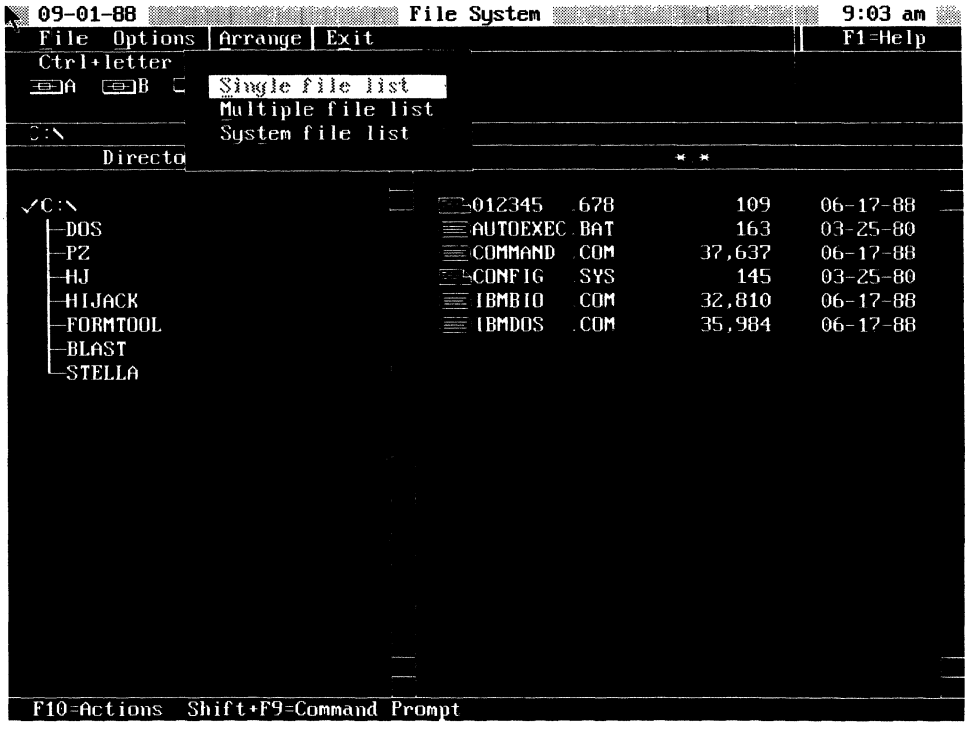


Figure 5.29
DOS 4.0 Arrange Pull-Down Menu

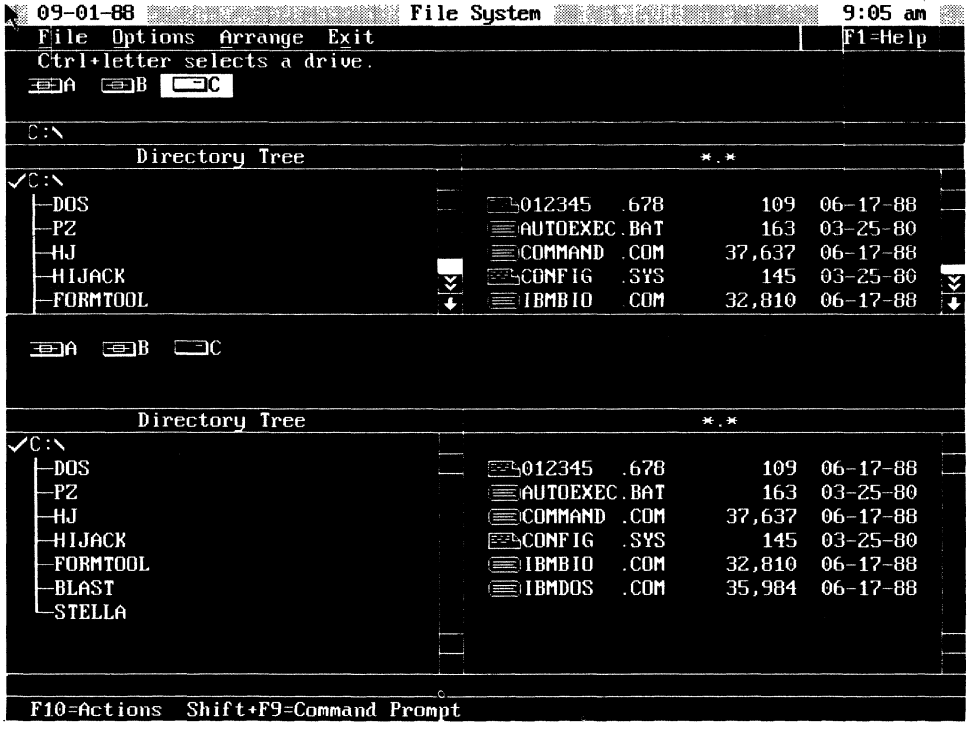


Figure 5.30
Initial Split Window

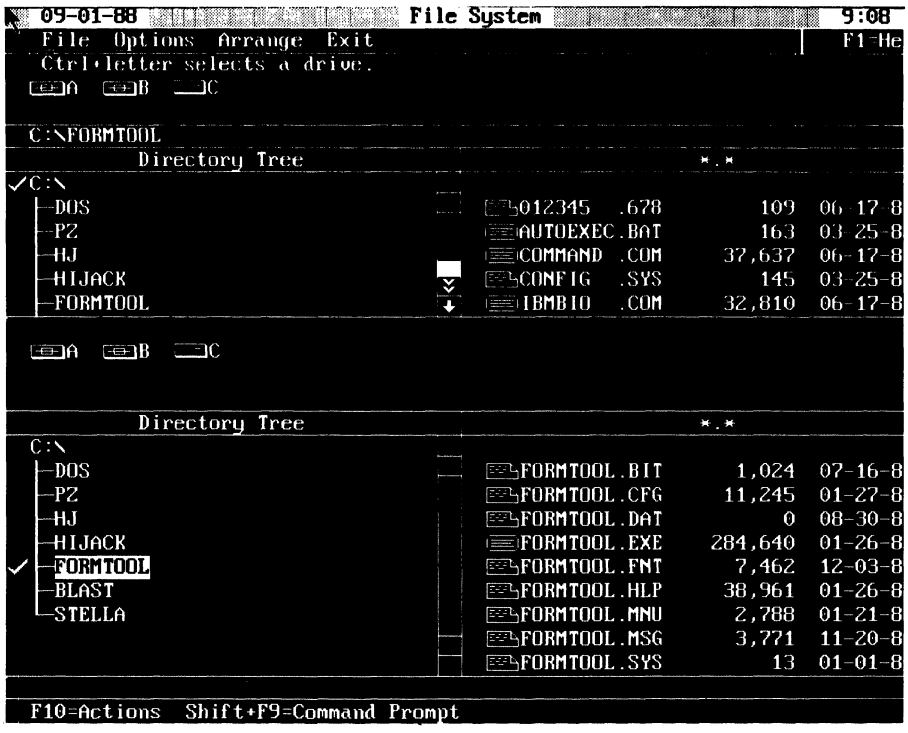


Figure 5.31
Split Window Showing Two Directories

range pull-down menu and checking the FORMTOOL directory in the bottom window's Directory Tree.

Note that you can use any one of the three file list actions in the Arrange menu to select a program to execute. Once you have located the desired file, you can select it by moving the selection cursor until it highlights the filename. Then, pressing the **Space bar** selects the file. At this time, you can press the **F10** key to select the File item in the action bar. Pressing **Down arrow** displays the File pull-down menu, the first selection item of which is Open (start). Then an Open File pop-up box is displayed, which appears in Figure 5.32. This pop-up box provides options to execute a program and to enter any optional parameters that may be associated with its execution.

As indicated in this section, the DOS Shell File System provides a similar, but enhanced, capability over DIR command prompt operations. Although with the File System you can view the contents of two directories at one time, sort directories five ways, and perform other enhanced operations, to use it effectively you should be familiar with wildcard characters.

COPY (Internal)

The COPY command duplicates one or more files on the same or a different diskette or directory/drive. If you copy files to the same diskette, you must assign the copy a different name from the original, unless you have created multiple directories to store

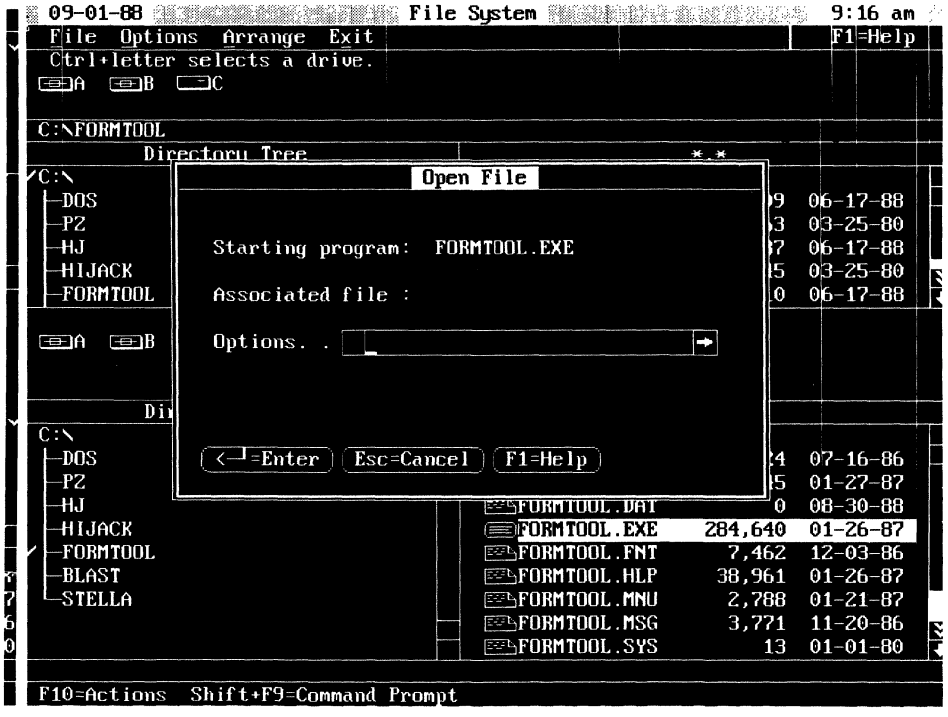


Figure 5.32
Using the Open File Pop-Up Box to Start a Program

them. If you copy files to a different subdirectory on the same fixed disk or diskette or to a different fixed disk or diskette, you can use the same name for both the original and the copied file.

The general format of the COPY command is

```
COPY[/A][/B][d:][path][filename[.ext]][/A][/B]
[d:][path][filename[.ext]][/A][/B][/V]
```

Although at first glance the general format of the command appears formidable, a short discussion of the /A and /B parameters should help you revise the format into a more manageable syntax for a majority of your file-copying operations. The /A and /B parameters control the amount of data to be processed by the COPY command. When used with a source file, the /A parameter causes the file to be treated as an ASCII (text) file. This means that all data in the file up to but not including a Ctrl+Z (end-of-file) character is copied.

If you use the /B parameter with a source file, the entire file is copied. If the /A parameter is used with the target file specification, a Ctrl+Z character is added as the last character of the file, whereas the use of the /B parameter does not add an end-of-file character (Ctrl+Z) to the file.

Normally, you can accept the default value of /A when *concatenation* (the process of joining files) is being performed during a COPY operation or the default value of /B when concatenation is not being performed. In certain situations, a communications

program may add a Ctrl+Z to a file being downloaded to a computer. If this occurs, the COPY command only copies a portion of that file, up to but not including the inadvertent end-of-file character. If you want to duplicate this type of file, use the /A parameter in the COPY command, with the source file specification.

Because a very large majority of COPY command operations can use the default value of the /A and /B parameters, the format of that command becomes more manageable, being rewritten as

```
COPY[d:][path][filename[.ext]][d:][path][filename[.ext]][/V]
```

The /V parameter causes DOS to verify that information copied is recorded properly. Because the verification process requires DOS to read the copied data to compare it with the original copy on a sector-by-sector basis, the /V switch causes the COPY command to operate slowly.

Both the ? and * wildcard (global) characters can be included in the filename and extension parameters of both the source and target files. When either global character is used in the source file specification, the names of the files are displayed as the files are copied. This is illustrated in Figure 5.33.

At the top of that figure, a wide directory listing of a diskette in drive B is shown. Note that there are four files on that disk. Next, the COPY command is entered to copy all files whose extension is .SU from the diskette in drive A to the diskette in drive B. Once this COPY command is entered, DOS displays the name and location of each file as it is copied. After all files matching the source filename with its global character are copied, DOS summarizes the COPY operation by displaying the message

```
3 File(s) copied
```

DOS always gives a numeric value that denotes the number of files that it copies onto the target medium.

Figure 5.33 Using a Global Character in a COPY Command

```
A>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE
        4 File(s)     80896 bytes free

A>COPY A:*.SU B:
A:DEMOLINE.SU
A:DEFAULT.SU
A:TYMNET.SU
        3 File(s) copied

A>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TYMNET   SU
        7 File(s)     77824 bytes free

A>
```

In the lower third of Figure 5.33, another wide directory listing of the contents of drive B appears. Note that seven files are now on the diskette in drive B, as a result of the COPY operation.

Physical Versus Logical Drives

The COPY command performed in Figure 5.33 was conducted on a PS/2 with two diskette drives. If your computer has only one diskette drive and you attempt to copy a file from one diskette to another, DOS treats the single physical drive as two logical drives. In such circumstances, DOS prompts you when to change diskettes, as illustrated by the following example.

```
A>COPY PAY.JLY B:PAY89.JLY

Insert diskette for drive B: and strike
any key when ready

        1 File(s) copied
```

Table 5.13 lists five examples of the use of the COPY command and their operational result. Note that under DOS 4.0 the prompt `A>` is `A:>` and the `C>` prompt is `C:\DOS>`, unless you use the DOS PROMPT command to change the prompts. The operational result of each command intentionally excludes mentioning that data is copied to the default subdirectory, because the use of the COPY command with a hierarchical directory structure is covered in detail in Chapter 6. That chapter examines how you can organize your fixed disk or diskette into subdirectories and place groups of related programs and data files in specific subdirectories.

Using Reserved Names

By using reserved device names in the COPY command, you can direct files to the printer, use the computer as a typewriter, and perform other interesting, unconventional

Table 5.13
COPY Command Examples

COPY Command	Operational Result
A>COPY B:*.* C:	Copies all files on the diskette in drive B to drive C.
A>COPY PAY.BAS C:	Copies the file PAY.BAS from the default drive (drive A) to drive C.
C>COPY A:*.BAS B:	Copies all files on the diskette in drive A whose extensions are .BAS to the diskette in drive B.
C>COPY A:*.BAS	Copies all files on the diskette in drive A whose extensions are .BAS to the default drive (drive C).
A>COPY PAY.BAS C:PAY89.BAS	Copies the file PAY.BAS from the default drive (drive A) to drive C and renames the file PAY89.BAS.

operations. Several examples of the use of reserved device names within a COPY command are

```
A>COPY PAY.DAT LPT1:

C>COPY CON: RUN.BAT

C>COPY PAY.BAS CON:
```

The first example routes the file PAY.DAT on the diskette in drive A to the printer attached to the LPT1 port.

In the second example, the console (CON) is used as the source device, and the file RUN.BAT is used as the target device. After you enter this COPY command, DOS routes keyboard input (from CON) to the file RUN.BAT, which DOS creates on drive C. This example illustrates one method by which you create batch files. To terminate keyboard input, you can either press the **F6** key or the **Ctrl+Z** multikey combination to generate an end-of-file character.

In the third example, the contents of the file PAY.BAS are routed to the console (CON) and are displayed on the screen.

A word of caution is in order when you copy files to the console or printer. Only files stored in an ASCII format are meaningful when printed or displayed with the COPY command. This is because the COPY command does not interpret and act on control codes embedded by word processors and other application software in a file. Thus, using the COPY command to print or display a file created with a word processing program that contains control codes for italics, underlines, and other attributes, or a BASIC program stored in a tokenized format, produces unexpected results in a display or printout. Printing or displaying the contents of such a file usually results in displaying or printing special characters that are not meaningful. This situation is illustrated by the use of the COPY command to print the contents of a small BASIC program whose output is shown in Figure 5.34.

DOS Shell COPY Operations

The File System of the DOS Shell can copy files. First, select the file you wish to copy. Similar to all file selection operations, you can use the **Tab** key to move the selection cursor to the filename area. Then use **Up** and **Down arrow** to move the selection cursor onto the filename you wish to copy. Then press the **Space bar**, which highlights the icon to the left of the filename.

Once the file or files you want copied are highlighted, you must select the Copy option from the File pull-down menu.

To display the File pull-down menu, press **Tab** to move the selection cursor to the File item in the action bar. Then press **Down arrow** to display the File pull-down menu, illustrated in Figure 5.35.

Figure 5.34
Printing a Non-ASCII File

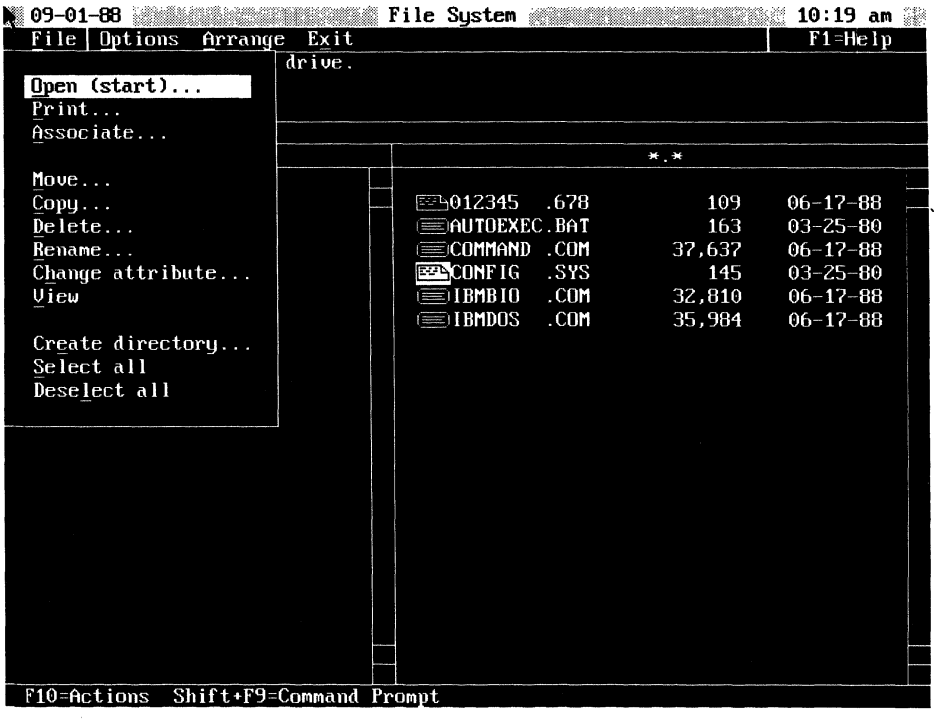


Figure 5.35
DOS 4.0 File Pull-Down Menu

After you have selected a file or set of files and accessed the File pull-down menu, you can move the selection cursor to the Copy option. Then press **Enter** to display the Copy File pop-up menu, shown in Figure 5.36. Note that the From box automatically contains the name of the selected file, whereas the To box displays the current drive and current directory, denoted as C:\ in this example, with the cursor positioned under the letter C. At this time, you can accept the current drive and current directory as the destination to which to copy the file, or you can enter a new location. Then you can enter the same filename or a different filename, depending on whether you changed the file's location and your naming requirements. You cannot have two files with the same name in the same directory; thus, if for example you want another copy of CONFIG.SYS in the root directory of drive C, you might name the copy CONFIG.OLD.

COMP (Compare Files) (External)

The COMP (compare files) command compares the contents of two files. This command should not be confused with the DISKCOMP command, which compares the contents of two entire diskettes.

The format of the COMP command is

```
[d:][path]COMP
[d:][path][filename[.ext]][d:][path][filename[.ext]]
```

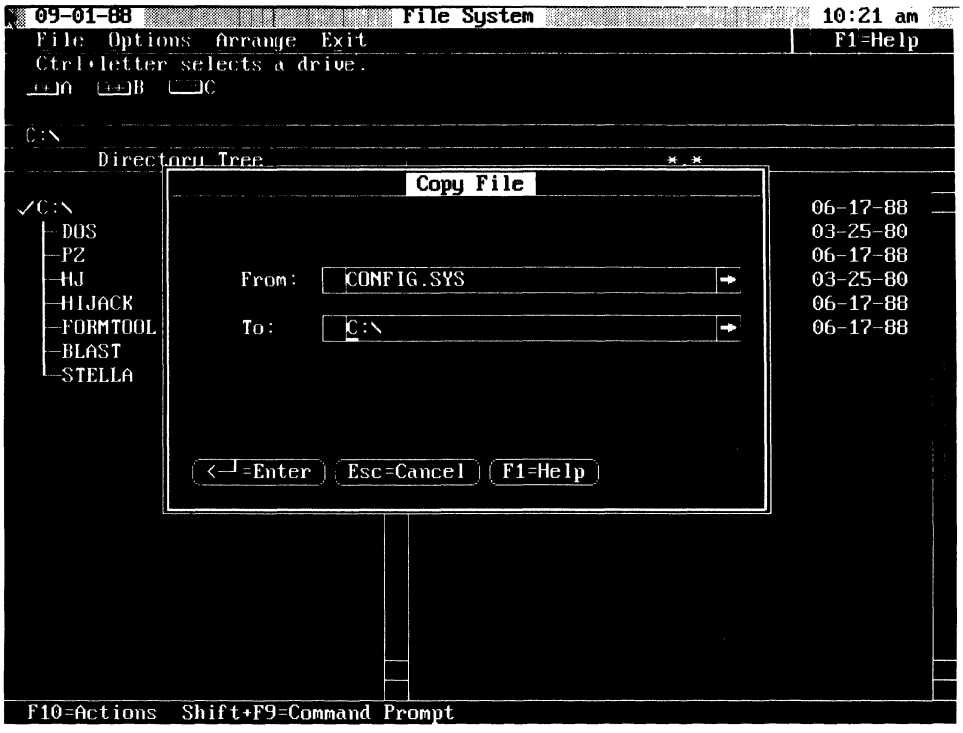


Figure 5.36
DOS 4.0 Copy File Pop-Up Box

Similar to other external commands, the device identifier (d:) and path preceding the command name identify the location where the COMP command file resides.

The first set of parameters ([*d:*][*path*][*filename*[.*ext*]]) after COMP specifies the first file or set of filenames you want to compare with wildcard characters used to define a set of files. The second set of parameters defines the file or set of files the first file or first set of files will be compared against.

You can compare files on the same drive or different drives, in the same directory, or in different directories. If you omit the filenames or the second set of parameters when you enter the command, you will be prompted to enter them.

After a successful comparison, COMP displays the message

```
Files compare OK
```

When all comparisons between files are complete, COMP displays the message

```
Compare more files (Y/N)?
```

At this point, if you enter **Y** you are prompted to enter two new filenames. You can type **N** to terminate the COMP command.

If the files being compared do not match, an error message is displayed. This error message indicates the location in hexadecimal notation where a mismatch occurred, as well as the contents of the bytes in each file where the mismatch occurred. After 10

unequal comparisons, COMP assumes that further comparison is of no use and displays the following message:

```
10 Mismatches - ending compare
```

Table 5.14 contains three examples of the use of the COMP command and the operational result of each command line entry.

DISKCOPY (Copy Diskette) (External)

The DISKCOPY (copy diskette) command duplicates an entire diskette or just the first side of a diskette. This command also examines the target diskette to determine whether it was previously formatted, and, if not, the command will format the target diskette with the same number of sides and sectors per track as the source diskette. The format of the DISKCOPY command is

```
[d:][path]DISKCOPY [d:[d:]][/1]
```

The first drive specifier after the command name identifies the source drive, whereas the second drive identifier indicates the target drive. If you specify the /1 parameter, DISKCOPY copies only the first side of the diskette, even if you use a double-sided diskette.

If you have only one diskette drive on your computer, DOS prompts you when to insert the target and source diskettes at the appropriate times. If you use the command without entering drive specifiers, a single-drive copy operation will be performed using the default drive. Similarly, if you specify the same drive for source and target diskettes a one-drive copy operation will be performed. Table 5.15 illustrates three examples of the use and operational results of the DISKCOPY command. Note that this command can only be used to copy similar capacity diskettes. Thus, you can use this command to copy a 720K byte double-sided diskette to another 720K byte diskette or a 1.44M byte double-sided diskette to another 1.44M byte double-sided diskette. However, you

Table 5.14
COMP Command Examples

COMP Command	Operational Result
A>COMP PAY B:PAY.DAT	Uses the COMP command file on the default drive (drive A) to compare the contents of the file PAY on drive A to the contents of the file PAY.DAT on drive B.
A>C:COMP A:PAY B:PAY.DAT	Uses the COMP command file on drive C to compare the contents of the file PAY on drive A to the contents of the file PAY.DAT on drive C.
C>COMP A:*.DAT C:	Uses the COMP command file on drive C to compare the contents of all files on drive A whose extensions are .DAT to files on drive C that have that extension.

Table 5.15
DISKCOPY Command Examples

DISKCOPY Command	Operational Result
C>**DISKCOPY A: B:**	Uses the DISKCOPY command file on drive C to copy the contents of the diskette in drive A to the diskette in drive B.
B>**C:DISKCOPY B: A:**	Uses the DISKCOPY command file on drive C to copy the contents of the diskette in drive B to the diskette in drive A.
A>**DISKCOPY**	Uses the DISKCOPY command file on drive A to perform a one-drive copy operation. DOS prompts you when to insert the source and target diskettes in that drive.

could not copy a 720K byte diskette to a 1.44M byte diskette or a 1.44M byte diskette to a 720K byte diskette.

DOS Shell Disk Copy Operations

You can perform a disk copy operation with the DOS Shell by selecting `DOS Utilities` from the Start Programs Main Group shown previously in Figure 5.10. This displays the DOS Utilities screen (see Figure 5.17). Move the selection cursor to `Disk Copy` and press **Enter** to display the Diskcopy Utility pop-up box on the DOS Utilities screen.

Figure 5.37 illustrates the Diskcopy Utility pop-up box. Note that the default entries in the `Drives` box are `a:` and `b:`, with the cursor positioned under `a`. You can accept either or both drive designators for the source and destination drives, or you can enter a different source or destination drive designator.

DISKCOMP (Compare Diskette) (External)

The DISKCOMP (compare diskette) command compares the contents of similar capacity diskettes. Normally, this command is used to verify the operation of a previously performed DISKCOPY command. The format of the DISKCOMP command is

`[`*d*`:][`*path*`]DISKCOMP[`*d*`:[`*d*`:]][/1][/8]`

Like all external commands, the optional drive identifier (d:) and path preceding the command name indicate where the DISKCOMP command file resides. If you do not specify a location, the default drive and current directory are used to locate the command file.

If the optional /1 parameter is included, DISKCOMP compares only the first side of the diskettes in the specified drives, even if the diskettes are double-sided. The /8 parameter instructs the command to compare diskettes on an 8-sectors-per-track basis, even if the first diskette contains 9 or 15 sectors per track.

Similar to the DISKCOPY command, you can enter the DISKCOMP command without drive parameters or with the same drive parameters to perform a one-drive comparison. When this occurs, you are prompted when to swap diskettes.

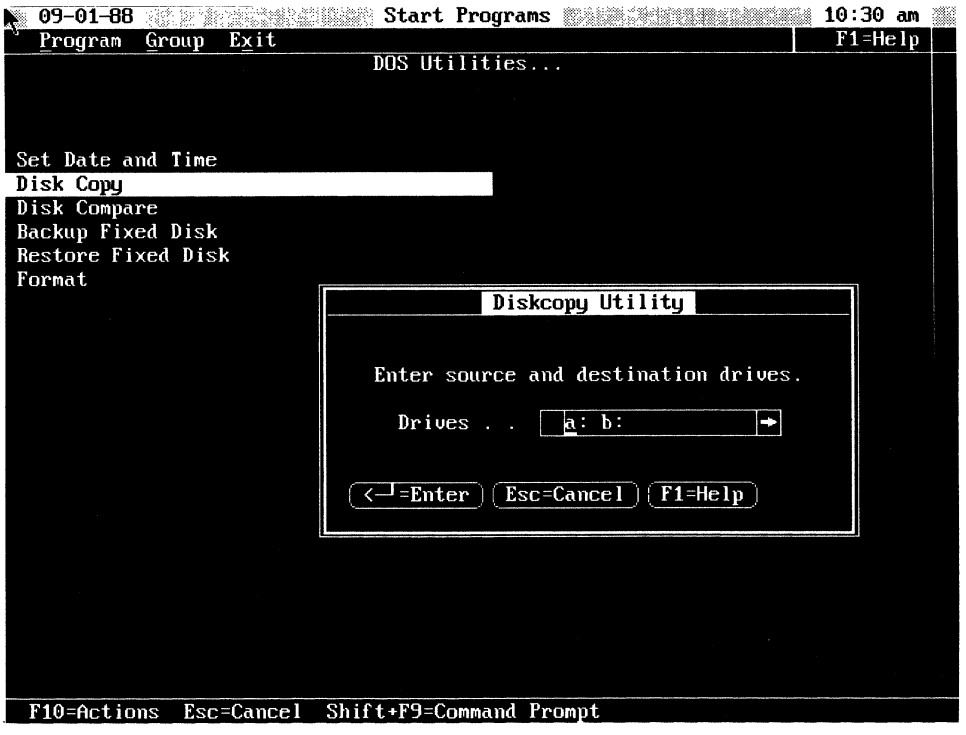


Figure 5.37
DOS 4.0 DISKCOPY Utility Pop-Up Box

When the DISKCOMP command is executed, all tracks on the source and target diskettes are compared on a track-by-track basis. If the tracks are not equal, a message indicating the track number of unequal tracks and the side where the mismatch occurred is displayed. Once the comparison is completed, the following message is displayed.

```
Compare more diskettes (Y/N)?
```

Entering **N** terminates the command operation. If you enter **Y**, another comparison is performed on the same drives after you answer the prompts to insert the appropriate diskettes. Table 5.16 lists three examples of the use and operational results of the DISKCOMP command.

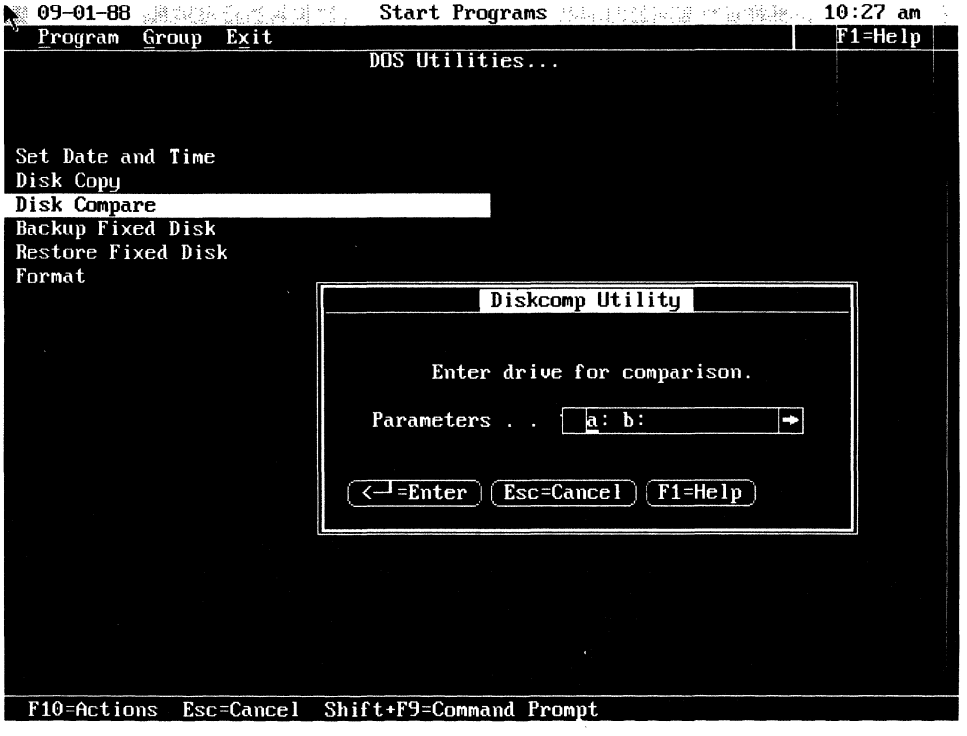
DOS Shell Disk Compare

Similar to the DOS Shell Disk Copy action, the Disk Compare option is selected from the DOS Utilities screen. Once you move the selection cursor over `Disk Compare` and press **Enter**, the DISKCOMP Utility pop-up box displays, as illustrated in Figure 5.38. Drives `a:` and `b:` are displayed as the default drives in the `Parameters` box. You can change one or both drive designators, or you can enter an optional parameter. Unfortunately, pressing the F1 key for Help refers you to IBM documentation concerning parameters used with this command. Thus, for this command—as well as other DOS commands performed under the DOS Shell—you should become familiar with the command line parameters unless you are willing to accept the command's default values.

Table 5.16
DISKCOMP
Command Examples

DISKCOMP Command	Operational Result
C>**DISKCOMP A: B:**	Uses the DISKCOMP command file on drive C to compare the contents of the diskette in drive A to the contents of the diskette in drive B.
B>**C:DISKCOMP B: A:**	Uses the DISKCOMP command file on drive C to compare the contents of the diskette in drive B to the contents of the diskette in drive A.
A>**DISKCOMP**	Uses the DISKCOMP command file on the diskette in drive A to perform a one-drive diskette comparison operation. DOS will prompt you when to insert the source and target diskettes in that drive.

Figure 5.38
DOS 4.0 DISKCOMP
Utility Pop-Up Box



DELETE and ERASE (Internal)

The DEL (delete) and ERASE commands are identical—they both have the same format and perform the same operation. Either command can be used to delete a specified file or group of files, the latter occurring when one or more global (wildcard) characters are used in the command line.

The difference between the two commands is the command name, with DEL used for delete and ERASE used for erase. The common format of these two commands is

$$\left\{\begin{matrix}\text{DEL}\\\text{ERASE}\end{matrix}\right\}\ [d:][path]filename[.ext]$$

Both DEL and ERASE can erase all files except hidden files, such as IBMBIO.COM and IBMDOS.COM. Figure 5.39 illustrates the use of the ERASE command using the asterisk wildcard to erase all files with the extension .SU from the diskette in drive B. At the top of Figure 5.39 is a wide directory listing, which indicates that there are seven files on the diskette in drive B, including three files whose extensions are .SU. In the middle of Figure 5.39, the ERASE command was used with an asterisk in the filename position. This command tells DOS to erase all files whose extension is .SU on the diskette in drive B. At the bottom of Figure 5.39, another wide directory listing of the diskette in drive B appears. Note that the three files with the extension .SU are no longer on the diskette.

In actuality, the use of DEL or ERASE does *not* physically remove the files from the diskette. When you use either command, an appropriate bit in the disk's File Allocation Table (FAT) is reset to inform DOS that previously used storage for that file can now be used to store other data. This is why many commercially available utility programs can recover previously erased files. If you should inadvertently erase a file, do not save any additional information on the disk before you use a data recovery program. This is because saving a file on a disk where a file was previously inadvertently erased may overwrite all or a portion of the sectors of the erased file with the new file. When this occurs, the previously erased file or a portion of that file is permanently lost.

Table 5.17 contains three examples of the use of the ERASE command. Each of these examples is applicable to the DEL command, because DEL has the same format and operational result as ERASE.

If you use a filename that does not exist in an ERASE or DEL command, DOS returns the message

```
File not found
```

Figure 5.39 Using the ERASE Command

```
A>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TYMNET   SU
        7 File(s)     77824 bytes free

A>ERASE B:*.SU

A>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE
        4 File(s)     80896 bytes free
```

Table 5.17
ERASE Command Examples

ERASE Command	Operational Result
A>**ERASE B:*.***	Deletes all files in the current directory on the diskette in drive B.
A>**ERASE PAY.***	Deletes all files named PAY in the current directory on the default drive (drive A) regardless of their extension.
B>**ERASE C:*.BAS**	Erases all files whose extension is .BAS that are located in the current directory on drive C.

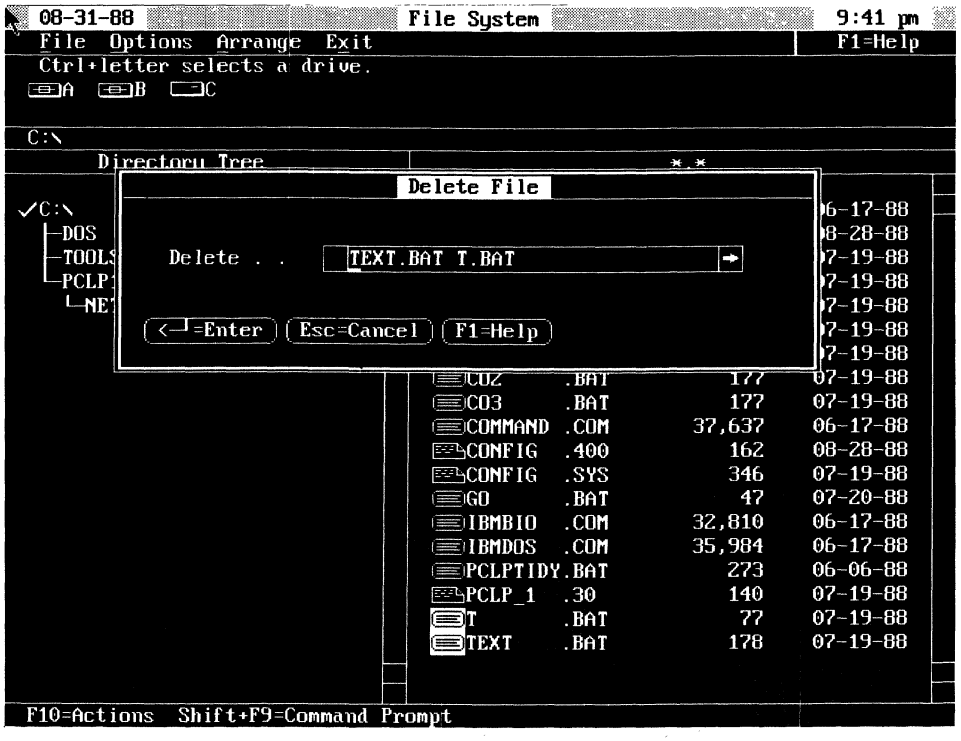


Figure 5.40
Initial DOS 4.0 Delete File Pop-Up Box

This message is applicable if you enter a specific filename and extension in an ERASE or DEL command that does not exist or if you use one or more global characters in the command and DOS fails to find a file that matches the entered specification.

DOS Shell Delete Operations

Similar to all file-related operations in the DOS Shell, you must first select a file or set of files. When operating with the DOS Shell, you can use the `Delete` option of the File pull-down menu (illustrated in Figure 5.35) to display the Delete File pop-up box. It is shown in Figure 5.40.

If you examine the bottom of Figure 5.40 carefully, you will note the icons for the files T.BAT and TEXT.BAT are highlighted; these files have been selected for deletion. This is why the `Delete` box displays the names of both files.

If you accept the default setting for `Confirm on delete` in the File Options pop-up box (see Figure 5.28), you are prompted for each file you want to delete. Figure 5.41 shows the options that enable you to skip the current file or delete it.

RENAME (Internal)

The RENAME command changes the name of a file or group of files, as well as their extensions. The format of this command is

$$\left\{\begin{matrix}\text{RENAME}\\\text{REN}\end{matrix}\right\} \; [d:][path]filename[.ext]filename[.ext]$$

As indicated in the format of the command, you can use either the full RENAME keyword or its abbreviation REN in a command line. The optional drive identifier (d:) and path following the command keyword indicate the location of the file or group of files whose name and/or extension are to be changed. The first filename and extension denote the present name and extension of a file or group of files, with the latter represented by the use of one or more wildcard characters. The second filename and extension in the command line denote the new name and/or extension of the file or group of files.

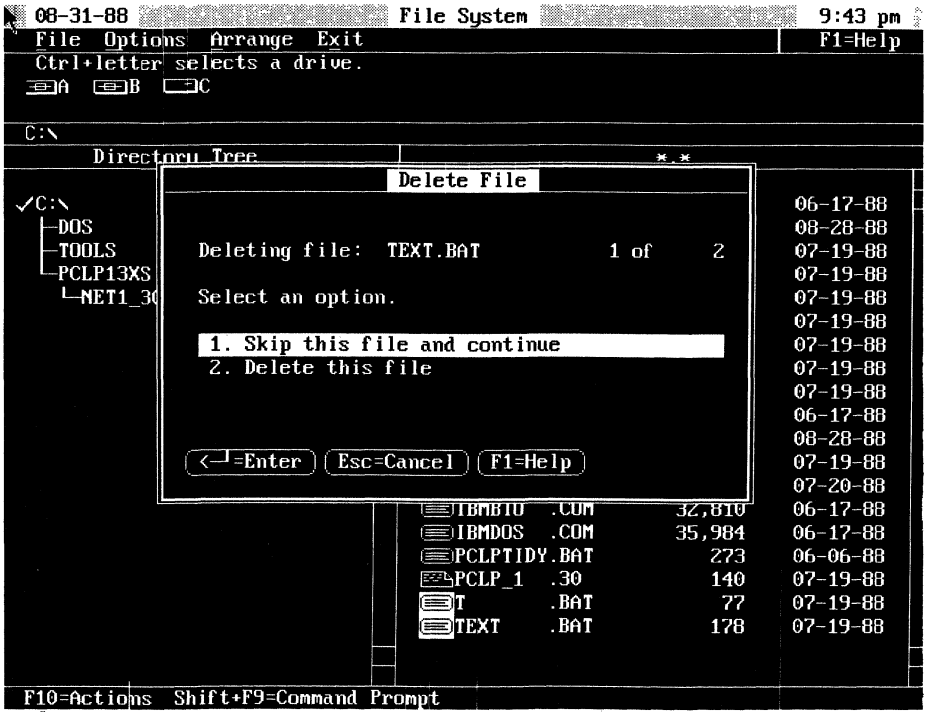


Figure 5.41
Using the DOS 4.0 Delete File Pop-Up Box

Figure 5.42 illustrates the use of the RENAME command to change the file named TYMNET.SU to TELENET.SU. At the top of that illustration, a wide directory listing of the contents of drive B appears. The middle of the figure shows that the RENAME command has been entered to change the filename of the TYMNET file shown in the directory. At the bottom of Figure 5.42, a second wide directory listing verifies that the name of the file has been changed.

Table 5.18 contains three examples of the use of the RENAME command and the operational result for each command. If you attempt to RENAME a file with a name that already exists in the current directory or if the file does not exist, DOS displays the following message:

```
Duplicate file name or file not found
```

DOS Shell RENAME Operations

Similar to all DOS Shell file operations, you must select a file prior to invoking the RENAME operation. Once you select a file or set of files, you can either press **F10** to

Figure 5.42
Using the RENAME Command

```
C>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST     EXE     INDEX     EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TYMNET    SU
        7 File(s)     77824 bytes free

C>RENAME B:TYMNET.SU TELENET.SU

C>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST     EXE     INDEX     EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TELENET   SU
        7 File(s)     77824 bytes free

C>
```

Table 5.18
RENAME Command Examples

RENAME Command	Operational Result
A>**RENAME *.SU *.DAT**	Changes the name of all files on the diskette in the default drive (drive A) whose extension is .SU to the extension .DAT.
C>**RENAME B:PAY.JUN JUNE.PAY**	Changes the name of the file PAY.JUN on the diskette in drive B to JUNE.PAY.
C>**RENAME PAY.JUN JUNE.PAY**	Changes the name of the file PAY.JUN on the default disk (drive C) to JUNE.PAY.

move the selection cursor over File or press **Tab** one or more times, depending on the current location of the selection cursor. Then, press **Down arrow** to display the file pull-down menu, which is the same as the menu illustrated in Figure 5.35.

Once you select Rename, the Rename File pop-up box will be superimposed on your screen. Figure 5.43 illustrates the Rename File pop-up box that contains the current filename, which was the filename previously selected. Note that if you select more than one filename, each filename is displayed and the box number increments from 1 of N to 2 of N, and so on, as you enter each new filename. Each time you enter a new filename and press **Enter**, the current file is renamed to the new filename.

TYPE (Internal)

The TYPE command provides convenient displays of ASCII file contents. The format of this command is

TYPE [*d*:][*path*]*filename*[.*ext*]

Similar to other commands that operate on files, the optional drive specifier (d:) and path indicate the location of the file whose contents are to be displayed. If not included, the default drive and current directory on that drive are used to locate the specified file.

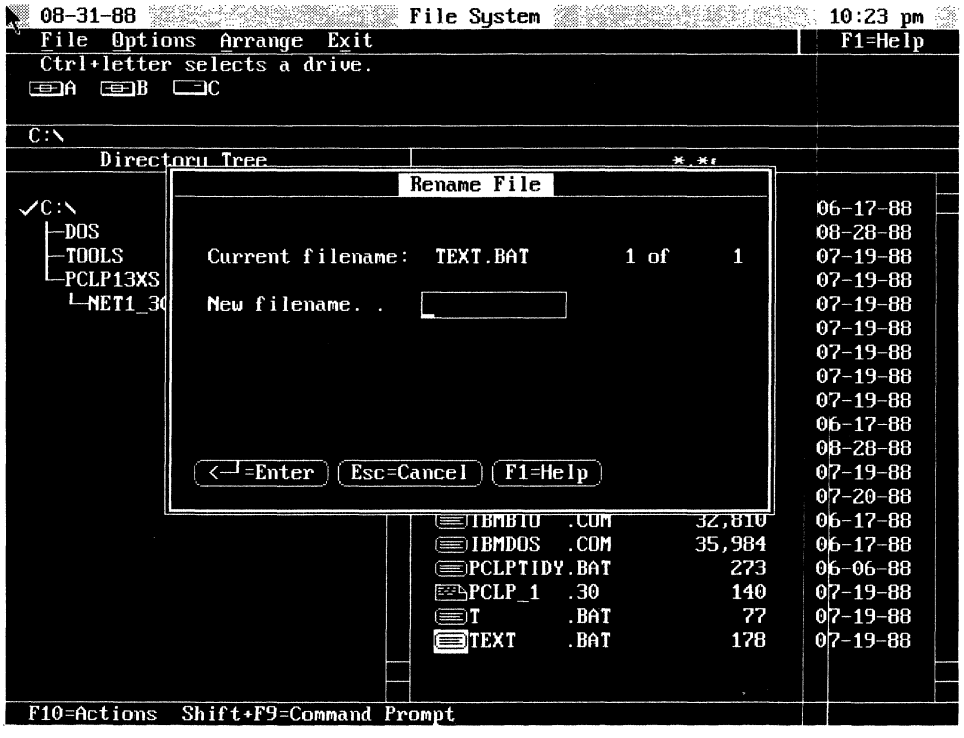


Figure 5.43
Using the DOS 4.0 Rename File Pop-Up Box

For lengthy files that would normally scroll off the screen, you probably prefer to obtain a hard copy of their contents. To do so, press the **Ctrl+PrtSc** multikey combination prior to entering the TYPE command. This causes all I/O to be automatically logged to the printer. Another method to direct the output of the TYPE command to the printer is output redirection. The concept of output redirection is covered in Chapter 9.

Table 5.19 illustrates the use of three examples of the TYPE command. The legibility of the file being displayed depends on the method used to store the contents of the file on disk. If the file was created by an application program, such as a word processing file, it should be stored in ASCII format to use the TYPE command effectively. Otherwise, embedded control codes used to define such characteristics as italics, bold print, and so on, are displayed, using their ASCII value. This normally creates a "jumbled" display, because control code characters represent interesting graphics for display purposes that have no relation to their use within an application program. Similarly, BASIC programs stored in a tokenized format and object program files also appear unreadable, because they contain nonalphanumeric characters.

DOS Shell VIEW Command

A more sophisticated version of the TYPE command is included in the DOS Shell in the View function. The View function is contained in the File item pull-down menu, illustrated in Figure 5.35.

As for all file reference operations, you must first select a file. Then, you can select the View function from the File pull-down menu, and a File View screen displays. It is similar to that illustrated in Figure 5.44.

Figure 5.44 shows the contents of the AUTOEXEC.BAT file that was previously selected. As indicated in the instructions displayed, you can press the **PgUp** and **PgDn** keys to scroll through the contents of a file. This feature is not available with the TYPE command, and the capability significantly improves your ability to view the contents of large files. Another improvement incorporated in the View function is the ability to toggle the display of the contents of a file from Hex (hexadecimal) to ASCII by pressing the F9 key, a feature valuable for debugging programs.

LABEL (External)

The LABEL command creates, changes, or deletes a previously created volume label. The volume label can consist of 1 to 11 characters and serves as an identifier for a diskette or for a fixed disk. The format of the LABEL command is

Table 5.19
TYPE Command Examples

TYPE Command	Operational Result
A>**TYPE READ.ME**	Displays the contents of the file READ.ME residing on the default (drive A) diskette drive.
C>**TYPE B:UPDATE.DOC**	Displays the contents of the file UPDATE.DOC residing on the diskette in drive B.
B>**TYPE A:INFO**	Displays the contents of the file INFO residing on the diskette in drive A.

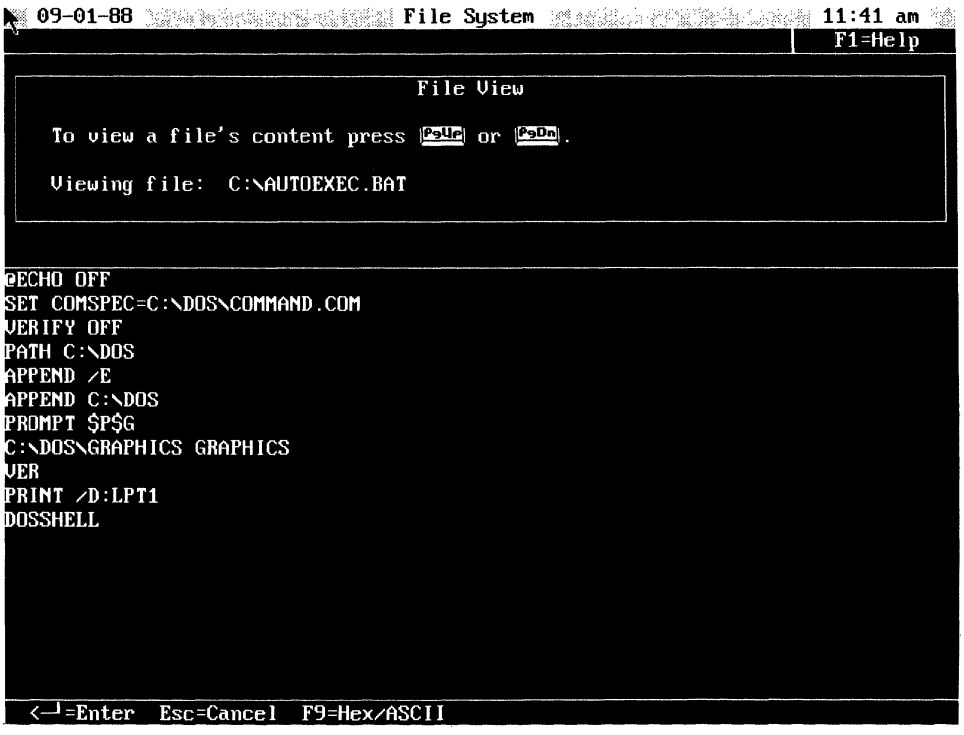


Figure 5.44
Using the DOS 4.0 View Facility

```
[d:][path]LABEL [d:][volume label]
```

The optional device identifier (d:) and path preceding the command name identify the location where the LABEL command file resides. If you do not specify a location, the default drive and current directory are used to locate the command file. The optional drive identifier following the command name identifies the drive on which the volume label will be created, changed, or deleted. If no drive is specified, the default drive will be used. The last optional parameter—the volume label—can be up to 11 characters in length. If this parameter is omitted, you are prompted with the following two messages:

```
Volume in drive X is xxxxxxxxxxx

Volume label (11 characters, ENTER for none)?
```

If you press the **Enter** key without typing a volume label, the following prompt message is displayed:

```
Delete current volume label (Y/N)?
```

If you type **Y** and press **Enter**, any previously assigned volume label is deleted.

Figure 5.45 illustrates the use of the LABEL command. At the top of that illustration a wide directory listing of the diskette in drive B was taken. Note that the volume label of the diskette is INVOICEJLY8. In the middle of Figure 5.45, the LABEL command

Figure 5.45
Using the LABEL Command

```
C>DIR B:/W

 Volume in drive B is INVOICEJLY8
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TELENET  SU
        7 File(s)     77824 bytes free

C>LABEL B:DATACOM

C>DIR B:/W

 Volume in drive B is DATACOM
 Directory of  B:\

COMMAND  COM     BLAST    EXE     INDEX    EXE     INSTALL  EXE     DEMOLINE SU
DEFAULT  SU      TELENET  SU
        7 File(s)     77824 bytes free
```

Table 5.20
LABEL Command Examples

LABEL Command	Operational Results
A>C:LABEL PAY	Uses the LABEL command file in the current directory of drive C to assign the volume label PAY to the diskette in the default drive (drive A).
A>LABEL B:PAY	Uses the LABEL command file in the current directory of the default drive (drive A) to assign the volume label PAY to the diskette in drive B.
C>LABEL A:	Uses the LABEL command file in the current directory of drive C to change the volume label of the diskette in drive A. Once the command line is entered the DOS prompt `Delete current volume label (Y/N)?` is displayed.

was used to change the volume label of the diskette in drive B to DATACOM. A second wide directory listing at the bottom of Figure 5.45 verifies the result of the previous use of the LABEL command.

Table 5.20 contains three examples of the use of the LABEL command and the operational result of each command line entry.

VOL (Volume) (Internal)

The VOL (volume) command displays the disk volume of a specified drive. The format of this command is

```
VOL [d:]
```

If you do not specify a drive, the volume label on the default drive is displayed. The following example illustrates the use of this command.

```
C>VOL B:

Volume in drive B is DATACOM

C>
```

Table 5.21 illustrates three examples of the use of the VOL command.

VERIFY (Internal)

The VERIFY command ensures that data written to a disk has been correctly recorded. Unlike the /V option in the COPY command that is in effect during the duration of that command, the VERIFY command remains in effect until you change its status by entering another VERIFY command or turn off power to your computer. The format of the VERIFY command is

```
VERIFY [{ON }]
       [{OFF}]
```

If you enter the VERIFY command without a parameter, the present state of the command is displayed. When you enter VERIFY ON, all data written to a disk is reread to verify that it was recorded without error. Due to the extra time required to read the data after it is written, VERIFY ON slows your computer's processing speed as it writes data to disk. The last command line option, VERIFY OFF, terminates any previous VERIFY ON command operation. The default value of VERIFY is OFF when DOS is initialized.

VER (Version) (Internal)

The VER (version) command, as its name implies, displays the DOS version number you are currently using. The format of this command is

```
VER
```

The following example illustrates the use of this command:

```
A>VER

IBM Personal Computer DOS Version 3.30

A>
```

Table 5.21
VOL Command Examples

VOL Command	Operational Result
A>**VOL**	Displays the disk volume label of the default drive (drive A).
A>**VOL C:**	Displays the disk volume label of the first fixed disk (drive C).
C>**VOL B:**	Displays the disk volume label of the diskette in drive B.

Utility Commands

The remainder of this chapter examines the operation and use of eight DOS commands that perform utility functions. Although these commands are not as commonly used as the commands previously covered in this chapter, each performs a valuable function you may find helpful.

ASSIGN Drive (External)

The ASSIGN command routes disk I/O requests for one or more drives to other drives. The format of this command is

```
[d:][path]ASSIGN [x[=]y[. . .]]
```

Like all external commands, the optional drive specifier and path preceding the command identify the location of the command file. The x parameter in the command specifies the drive to which current disk I/O requests are routed. The y is replaced by the drive letter you want disk I/O requests routed to. If you do not specify x and y parameters, the command resets all drive parameters, resulting in the resumption of normal drive assignments.

Your usual use of the ASSIGN command is with programs that restrict file storage to certain drives. It is used prior to loading an application program that was written to perform I/O operations to a specific drive that you wish to change. Although a large majority of software now permits users to specify any disk for data storage, prior to the introduction of the PC XT many application programs restricted I/O operations to diskette drives A and B. Thus, if you are using a program with this limitation, the ASSIGN command enables you to store data files on your fixed disk.

In the example illustrated in Figure 5.46, the ASSIGN command assigns diskette drive B I/O references to the fixed disk, drive C. Next, the user obtains a wide directory listing of all files with the extension .BAT on drive B. By examining the result of the use of the DIR command, it is obvious that physical drive C has been used, because no diskette currently marketed could have over 21 million bytes of available storage capacity.

CHKDSK (Check Disk) (External)

The CHKDSK (check disk) command helps you to analyze the directories, files, and the File Allocation Table (FAT) on a disk or diskette. The result of the execution of

Figure 5.46 Using the ASSIGN Command

```
C>ASSIGN B=C

C>DIR B:*.BAT/W

 Volume in drive B is DATACOM
 Directory of  B:\

PMTUTOR  BAT    AUTOEXEC BAT    GOMOUSE  BAT    BEEP     BAT    DESIGN   BAT
COMPRESS BAT    TESTCOM  BAT
        7 File(s)  21729280 bytes free

C>
```

this command is a display of information concerning the available and used disk space of the designated drive and the computer's total and free memory status. The format of this command is

```
[d:][path]CHKDSK [d:][path][filename[.ext]][/F][/V]
```

The optional parameters preceding CHKDSK specify the drive and path to the directory that contains the CHKDSK command file. The /F parameter causes CHKDSK to fix any errors the program encounters in the disk's directory or FAT. The /V parameter generates a display of all files and their paths on the default or specified drive.

When you use the CHKDSK command, if you specify a filename the command displays the number of noncontiguous areas occupied by the file. If this number becomes very large due to the frequent use and modification of the contents of the file, you may want to consider "unfragmenting" the file. To do this, use the COPY command to copy the file to a newly formatted disk. Thereafter, disk file I/O times should be significantly faster, because DOS will be initially working with a file whose contents are stored in a contiguous location on the disk.

Figure 5.47 shows how to use the CHKDSK command under DOS 3.3 to check a diskette in drive B, as well as the number of noncontiguous blocks occupied by the file NEWINDEX.EXE located on the diskette in drive B.

Under DOS 4.0, the CHKDSK command and its parameters are used the same as under DOS 3.3. However, additional information concerning the data recorded on the disk is displayed. Figure 5.48 illustrates the use of the CHKDSK command under DOS 4.0. Note that when a fixed disk is checked, its volume serial number displays under this version of the operating system. In addition, for both fixed disks and diskettes, three lines of information concerning their *allocation units*—IBM's new name for clusters—are displayed.

As you examine Figure 5.48, note that there are 2048 bytes in each allocation unit, which means that data is recorded and retrieved in clusters of 2048 characters. Because there are a total of 10337 allocation units on the disk and each allocation unit contains 2048 bytes, multiplying the number of bytes in each allocation unit by the total number of allocation units on the disk yields the total disk space—21170176 bytes in the example illustrated in Figure 5.48.

Table 5.22 contains three examples of the use and operational results of the CHKDSK command.

Figure 5.47
Using the CHKDSK Command

```
C>CHKDSK B:NEWINDEX.EXE
Volume DATACOM      created Jun 21, 1988 8:13a

   362496 bytes total disk space
    53248 bytes in 3 hidden files
   244736 bytes in 7 user files
    64512 bytes available on disk

   655360 bytes total memory
   573792 bytes free

B:\NEWINDEX.EXE
   Contains 2 non-contiguous blocks.

C>
```

Figure 5.48
CHKDSK Display Under DOS 4.0

```
C:\DOS>chkdsk

Volume DOS400      created 03-25-1980 1:45a
Volume Serial Number is 2E61-08E9

  21170176 bytes total disk space
     71680 bytes in 3 hidden files
     16384 bytes in 7 directories
   3639296 bytes in 182 user files
      6144 bytes in bad sectors
  17436672 bytes available on disk

      2048 bytes in each allocation unit
     10337 total allocation units on disk
      8514 available allocation units on disk

    655360 total bytes memory
    543920 bytes free

C:\DOS>
```

Table 5.22
CHKDSK Command Examples

CHKDSK Command	Operational Result
A>**C:CHKDSK B:**	Uses the CHKDSK command file on the current directory in drive C to analyze the diskette in drive B.
A>**CHKDSK**	Uses the CHKDSK command file on drive A to analyze the diskette in that drive. The message `Bad command or file name` is displayed if the CHKDSK command file is not on the diskette in drive A.
C>**CHKDSK A:PAY.DAT**	Uses the CHKDSK command file on the current directory in drive C to analyze the diskette in drive A and the noncontiguous storage used by the PAY.DAT file on that diskette.

ATTRIB (Attribute) (Internal)

The ATTRIB (attribute) command displays or modifies file attributes for a single file, selected files in a directory, or all files in a subdirectory. The two attributes that can be set or displayed are the files read attribute and its archive bit. The format of this command is

$$[d:][path]\texttt{ATTRIB}\ \left[\left\{\begin{matrix}+\texttt{R}\\-\texttt{R}\end{matrix}\right\}\right]\ \left[\left\{\begin{matrix}+\texttt{A}\\-\texttt{A}\end{matrix}\right\}\right]\ [d:][path][filename[.ext]][/\texttt{S}]$$

The optional +R and -R parameters are used to set and remove the read-only attribute of the specified file. Similarly, the +A and -A parameters are used to set and turn off the archive bit of the specified file. When either of these parameters is included in the ATTRIB command, you must include a file specification in the command line. This

specification can include one or more wildcard characters in the filename and file extension fields.

The setting of the read-only attribute of a file is a valuable mechanism to prevent other users of your computer from modifying selected files. Figure 5.49 illustrates the use of the ATTRIB command. In the top portion of the figure, the +R parameter is included in the command to set the read-only attribute of the file BLAST.EXE located on drive B. In the middle of Figure 5.49, the user entered the ERASE command in an attempt to delete the file whose read-only attribute was previously set. Note the DOS message `Access denied` to the attempt to erase the file. This action occurred since the file's read attribute was previously set to read-only. This setting is confirmed by the use of the ATTRIB command without the R or A parameters in the lower portion of Figure 5.49. Here the line displayed after the ATTRIB command was entered denotes that the archive bit and the read-only attribute of the file are set.

If you include the optional /S parameter in the ATTRIB command, all files in the specified directory and any files in subdirectories under that directory are processed. Table 5.23 lists three examples of the use of the ATTRIB command and their operational results.

DOS Shell File Attribute Operations

Under the DOS Shell, you can view four types of file attributes—`a` for archive, `h` for hidden, `s` for system, and `r` for read-only. You can view the attributes associated with one or more files by first selecting the file. Then you can select the `Show information` item from the Options menu, which displays the files' attributes.

Figure 5.49
Using the ATTRIB Command

```
A>ATTRIB +R B:BLAST.EXE

A>ERASE B:BLAST.EXE
Access denied

A>ATTRIB B:BLAST.EXE
  A     R  B:\BLAST.EXE

A>
```

Table 5.23
ATTRIB Command Examples

ATTRIB Command	Operational Result
A>C:ATTRIB +R B:PAY.DAT	Uses the ATTRIB command file on drive C to set the read-only attribute of the file PAY.DAT on drive B.
A>ATTRIB C:PAY.DAT	Uses the ATTRIB command file on the default drive (drive A) to display the attribute settings of the file PAY.DAT on drive C.
C>ATTRIB +R -A PAY.DAT	Uses the ATTRIB command file on the default drive (drive C) to set the read attribute and reset the archive bit of the file PAY.DAT located on drive C.

Although you can display four types of attributes, you can only change three types—hidden, read-only, or archive. To change a file attribute, first select the file by moving the selection cursor to the filename area. Then, use **Up** and **Down arrow** to move the selection cursor over the first file whose attribute you want to change. Pressing the **Space bar** selects the file, highlighting the icon to the left of the filename. After you have completed your file selection, press either **F10** or **Tab** to position the selection cursor over `File` in the action bar. Then, press **Down arrow** to display the File pull-down menu, previously illustrated in Figure 5.35. After you move the selection cursor to `Change attribute` and press **Enter**, the Change Attribute pop-up box appears, as illustrated in Figure 5.50. As indicated by the entries in this pop-up box, you can change the attributes of files one at a time or you can change the attributes of all selected files at one time. Select the first option when you want to give different attributes to each of the selected files.

Once you select the method by which file attributes will be changed (one at a time or all at once), a new pop-up box is displayed that contains three entries—`Hidden`, `Read Only`, and `Archive`. You can change one or more attributes associated with a specific file or one or more attributes of all files by highlighting an item and pressing the **Space bar**. This displays a mark to the left of the attribute name. You can change more than one attribute by moving the selection cursor to other attribute items and pressing the **Space bar**. Finally, to effect the attribute changes press the **Enter** key. If `Hidden` is selected, the DIR command will not display information about the file. However, the

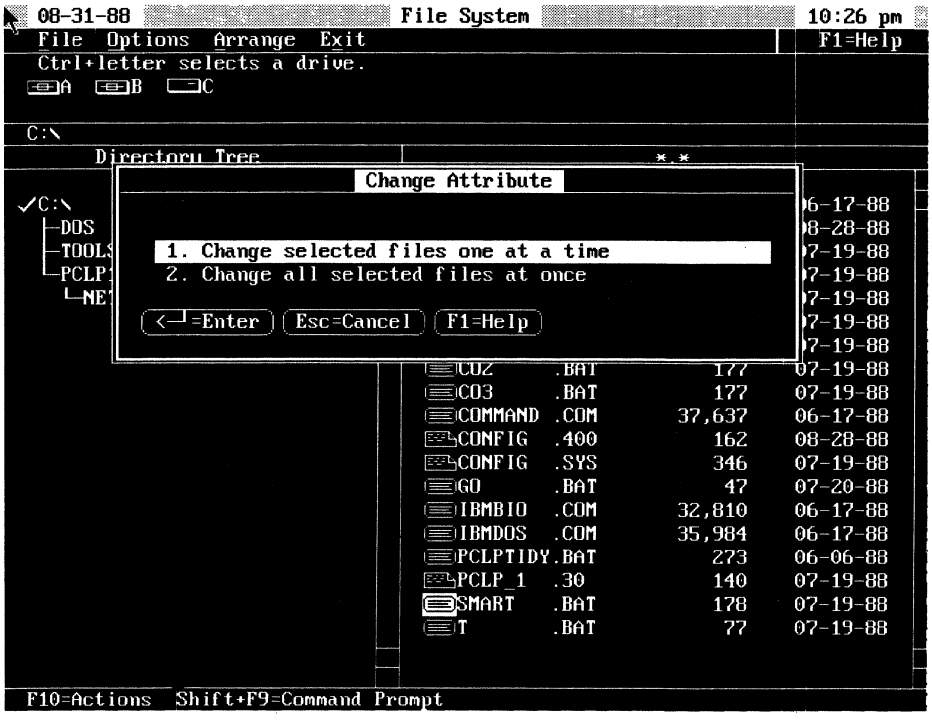


Figure 5.50
DOS 4.0 Change Attribute Pop-Up Box

filename will still appear under the DOS Shell list. Both `Read Only` and `Archive` choices function in the same manner as when the ATTRIB command is used.

GRAPHICS (External)

Under DOS 3.3, the GRAPHICS command provides a printed copy of a graphics display when your computer is placed in a color/graphics monitor adapter (CGA) video mode (and if your printer can accommodate graphics). Under DOS 4.0, screen printing of text and/or graphics was extended to support Enhanced Graphics (EGA) and Video Graphics Adapter (VGA) hardware. Through this command, a screen containing graphics *picture elements (pixels),* including any text formed by pixels, is printed on the specified graphics-compatible printer when you press the **PrtSc** key. The format of this command is

`[d:][path]GRAPHICS [printer type][/R][/B][/LCD]`

The two optional parameters that precede the command identify the location of the command file. Under DOS 3.3, the printer type option enables you to select one of six devices:

COLOR1	IBM PC Color Printer with black ribbon
COLOR4	IBM PC Color Printer with red, green, blue, black ribbon
COLOR8	IBM PC Color Printer with cyan, magenta, yellow, black ribbon
COMPACT	IBM PC Compact Printer
GRAPHICS	IBM Personal Graphics Printer and IBM ProPrinter
THERMAL	IBM PC Convertible Printer

Under DOS 4.0, the printer type GRAPHICSWIDE was added and the number of printers supported by the GRAPHICS printer type was considerably expanded. GRAPHICS printers that are now supported include the IBM 5152 Graphics Printer Model 2, the ProPrinter, ProPrinter II, ProPrinter XL with 8.5-inch wide paper, ProPrinter X24 and XL24 with 8.5-inch wide paper, the Pageprinter and the IBM Quietwriter II and Quietwriter III. Under the GRAPHICSWIDE printer type, the IBM Quietwriter II, ProPrinter XL, and ProPrinter XL24 with 13.5-inch wide paper are now supported. The GRAPHICS and WIDEGRAPHICS printer parameters provide compatibility with a large number of non-IBM printers. This is because the IBM Personal Graphics Printer sold with the original IBM PC was manufactured by Epson Corporation, and its printer control codes are used in many other vendor devices, including the IBM ProPrinter series. If you do not specify a printer type in the command, the default is the GRAPHICS printer due to the popularity of such devices with compatible printer control codes.

The /R parameter causes black to be printed as black and white as white. If you do not specify a setting, black is printed as white and white as black.

The /B parameter is required to print the background color and is only applicable to printer types COLOR4 and COLOR8. If this parameter is not specified, the background color is not printed.

When you invoke the GRAPHICS command, the resident size of DOS in memory is increased by the size of the program required to dump the pixels to the specified printer.

Note that all text on the screen, as well as the graphics picture elements, is printed on the printer when you invoke this command.

GRAFTABL (Load Graphics Table) (External)

The GRAFTABL (Load Graphics Table) command loads a table of data into your computer's memory that defines ASCII characters 128 through 255. Once GRAFTABL is invoked, ASCII characters 128 through 255 can be displayed when you use the color/graphics adapter video mode. The format of this command is

`[d:][path]GRAFTABL [country page code][/STATUS]`

where the country page code numbers for available countries are

Country Page Code Number	*Country*
437	United States (default value)
860	Portugal
863	Canada (French)
865	Norway and Denmark

Entering an appropriate country page code number in the command line causes DOS to load a table of data into your computer's memory. This table defines the display characteristics of ASCII characters 128 through 255. You can subsequently display these characters when your computer uses the color graphics adapter video mode. If you use the /STATUS parameter, the command displays the number of the currently selected country page code.

Table 5.24 illustrates three examples of the use of the GRAFTABL command. Note that this command can be used multiple times to change the current table of data in memory to represent a different country.

Table 5.24
GRAFTABL Command Examples

GRAFTABL Command	Operational Result
A>**C:GRAFTABL 860**	Uses the GRAFTABL command file in the current directory on drive C to load the table of graphics characters for the Portuguese code page.
.>**C:GRAFTABL /STATUS**	Uses the GRAFTABL command file in the current directory on drive C to display the number of the selected country code page.
C>**GRAFTABL 437**	Uses the GRAFTABL command file in the current directory on the default drive (drive C) to load the table of graphics characters for the US code page.

SYS (System) (External)

The SYS (system) command transfers the operating system files IBMBIO.COM and IBMDOS.COM to a disk with an empty directory or a disk that was previously formatted by the FORMAT command that included the /S or /B parameters in the command line. Either parameter provides space for the system files to be transferred to occupy the first two entries on the target diskette, permitting that diskette to be used as a start-up or boot diskette. This command is normally used to transfer a copy of DOS onto application program diskettes that are designed to use DOS but that cannot be legally sold with the previously mentioned IBM files. Once you transfer the system files to the application diskette, you can thereafter use the application diskette without first bringing up DOS. This command is especially useful if you have a one-diskette drive system. Otherwise, you have to do the "floppy shuffle," where after bringing up DOS you remove the DOS diskette so you have an available drive for the application diskette.

The format of the SYS command is

```
[d:][path]SYS d:
```

The drive identifier (d:) after the command keyword identifies the disk drive that the operating system files will be transferred to. The following example shows the use of this command to transfer the operating system files from a diskette in drive A to a diskette in drive B.

```
A>SYS B:
System transferred
```

If you attempt to transfer the system files to a diskette whose directory is not completely empty or to a diskette that was not formatted using a /S or /B parameter in the command line, the location where the system files must be located will be unavailable for use on the target diskette. To indicate this situation, DOS displays the following error message:

```
No room for system or destination disk
```

Table 5.25 lists two examples of the use of the SYS command and the operational result of each command line entry.

Table 5.25
SYS Command Examples

SYS Command	Operational Result
A>C:SYS B:	Uses the SYS command file on the current directory of drive C to transfer the system files to the diskette in drive B.
A>SYS B:	Uses the SYS command file on the current directory of the default drive (drive A) to transfer the system files to the diskette in drive B.

MEM (External)

The MEM command is new in DOS 4.0. This command can display the amount of used and unused memory, as well as programs currently in your computer's memory. The format of this command is

```
[d:][path]MEM [{/PROGRAM}]
              [{/DEBUG  }]
```

When used without any parameters, MEM produces a display containing a summary of used and unused memory, as illustrated in Figure 5.51. The command will display extended memory only if memory above 1M byte is installed and if an expanded memory driver was loaded. The /PROGRAM option causes all programs currently memory resident to be displayed as well as a summary of used and unused memory. If the /DEBUG option is included in the command line, a complete summary of information, including system device drivers, install device drivers, all programs currently in memory, and all used and unused memory, will be displayed. Refer to Chapter 8 for additional information concerning device drivers.

Table 5.26 illustrates two examples of the use of the MEM command and the operational result of each command line entry.

PROMPT (Set System Prompt) (Internal)

This command sets a new DOS prompt or resets the prompt to its default value. The format of this command is

```
PROMPT [prompt-text]
```

Figure 5.51 Using the DOS 4.0 MEM command

```
C:\DOS>mem

    655360 bytes total memory
    654336 bytes available
    543920 largest executable program size

    393216 bytes total extended memory
    393216 bytes available extended memory

C:\DOS>
```

Table 5.26 MEM Command Utilization Example

MEM Command	Operational Result
C:\DOS>**MEM**	Uses the MEM command file located in the DOS directory of drive C to display the amount of used and unused memory.
C:\DOS>**MEM /PROGRAM**	Uses the MEM command file located in the DOS directory of drive C to display all programs in memory as well as a summary of used and unused memory.

where prompt-text can contain metastrings in the form of $C and text that define the new system prompt. If you omit the optional prompt-text, the current prompt is reset to the normal DOS prompt. Table 5.27 lists the metastrings that you can include in the PROMPT command and summarizes their meanings.

To illustrate the use of the PROMPT command, enter the command **PROMPT $T $B NG** after the normal DOS prompt. As shown here, the metastring $T results in the display of the time, whereas the metastring $B generates the piping character (¦), and the metastring NG generates the default drive letter followed by the greater than sign.

```
C>PROMPT $T $B $N$G
16:27:35.81 ¦ C>
```

You can return to the normal DOS prompt by entering the command **PROMPT** with no characters. If you wish, you can include text in the PROMPT command, as indicated by the following example.

```
A>PROMPT COMMAND?$Q
COMMAND?=
```

The primary purpose for the inclusion of the PROMPT command in DOS was to provide a mechanism to structure a customized prompt to fit your organizational requirements. A secondary purpose of this command is to generate control codes through the use of the Escape character that can be included in batch files to perform predefined operations. Chapter 8 contains several examples of the use of the PROMPT command in batch files.

The AUTOEXEC.BAT file created by the DOS 4.0 INSTALL program also illustrates the use of the PROMPT command and explains why the DOS 4.0 prompt is initially

Table 5.27
Metastring Meanings

Metastring	Meaning
$	$ character
t	The time
d	The date
p	The current directory of the default drive
v	The DOS version number
n	The default drive letter
g	> character
l	< character
b	¦ character
q	= character
h	A backspace; the previous character is erased
e	The Escape character
-	The CRLF (Carriage Return Line Feed) sequence; causes the cursor to go to the beginning of a new line on the display

different from the DOS 3.3 prompt. In examining Figure 5.44, in which the AUTOEXEC.BAT file was displayed by the use of the DOS Shell function, note the line PROMPT PG. The inclusion of this PROMPT command in the AUTOEXEC.BAT file causes the current directory of the default drive ($P) followed by the > character ($G) to display as the system prompt. Thus, by removing or modifying this command you can automatically have your computer display a different system prompt whenever you perform a power-on or system reset operation.

6 / Fixed Disk Organization

This chapter focuses on the use and operation of DOS commands that enable you to use a fixed disk effectively. First, the chapter reviews the concept behind hierarchical directory structures. Next, you will examine several types of hierarchical directory structures so you can make an informed decision about which structure best satisfies your organizational requirements. Using the preceding information as a base, you can then explore DOS directory-related commands to construct a predefined directory structure. Once this is accomplished, the text explains the use of paths in DOS commands to load applications onto and operate them from a fixed disk. Although this chapter describes the DOS command prompt mode of operation, when applicable it also informs you when you can use the DOS Shell to perform equivalent operations.

Hierarchical Directory Structures

Until DOS Version 2.0, the IBM PC was limited to supporting a single directory on diskette storage. The IBM PC XT included a 10M byte fixed disk, which increased the storage capacity available to personal computer users more than thirtyfold. Whereas this additional storage capacity was welcomed by personal computer users, without a hierarchical directory structure it would have presented several problems that users had not heretofore encountered with the limited storage capacity of diskettes.

First, access to files on a fixed disk could become exceedingly slow, because the operating system might have to search a directory that could contain hundreds to thousands of files.

Second, from an organizational perspective, storing all program and data files under a common directory would tax the mental capability of most computer users. As an example, a user storing several hundred files on a fixed disk would have a difficult time remembering which shared data files various users had created, or even which data files were generated by a word processor, say, and which by a spreadsheet or database manager.

To alleviate these problems, the introduction of the IBM PC XT was accompanied by DOS Version 2.0, which included support for hierarchical directory structure operations. This feature, which remains part and parcel of the newest PS/2 systems, enables you to establish an almost infinite number of subdirectory structures that can be customized to meet both individual and organizational data storage requirements on fixed disks.

The top of the directory in a hierarchical directory structure is the *root directory.* This name is derived from the fact that a hierarchical directory structure is similar to an inverted tree. Like the root system of a tree in which nutrients flow from the root to branches, access to subdirectories flows through the root directory.

Entries in a hierarchical structure can consist of DOS files or other directories. For subordinate levels of directories, the entry is more accurately known as a *subdirectory.* Figure 6.1 illustrates the nesting capabilities of a hierarchical directory structure, showing how you can organize your storage media to place files in predefined directories.

Two subdirectories, SD1 and SD2, are illustrated in the figure. Subdirectory SD1 is located directly under the root directory, whereas subdirectory SD2 is nested under SD1. You can access a file or subdirectory by establishing a path through the directory structure to the desired file or subdirectories. Here the path is the route that informs DOS of the specific location on a storage media where files and subdirectories are located. Once you access a hierarchical directory structure, you can move up and down the structure by specifying an appropriate path name.

Directory Structures to Consider

Under DOS's hierarchical directory structure, when you use the FORMAT command DOS automatically creates a root directory on each diskette and fixed disk. The root directory for a specific device is indicated by the drive designator followed by a backslash. As an example,

```
C:\
```

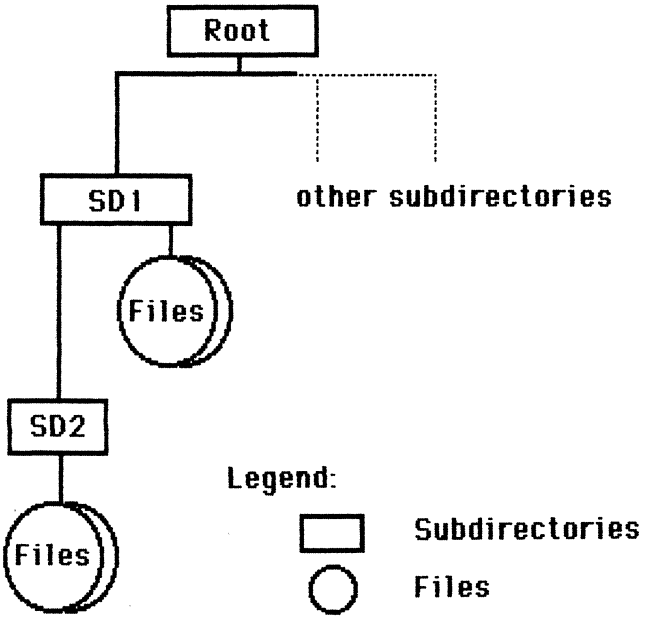


Figure 6.1
Nesting Subdirectories

indicates a current position at the top of the tree structure or the root directory of the fixed disk, whereas

```
A:\
```

indicates the root directory of a diskette in drive A.

Although you can use a hierarchical directory structure on diskettes, in most situations you will probably maintain one directory structure and locate files under a common root directory for this storage medium. This is because you will probably prefer to organize your data and program files by diskette, placing a word processing program and its data files on one diskette, a spreadsheet program and its data files on a second diskette, and so on. When you use a fixed disk, the large storage capacity of that device is best organized by structuring its storage. This structuring can take almost an infinite number of shapes due to the flexibility DOS provides in naming and nesting subdirectories. Two of the more general hierarchical directory structures you may wish to consider are illustrated in Figure 6.2.

The top portion of Figure 6.2 represents an application-oriented directory structure. In this type of directory structure, application programs and their data files are organized

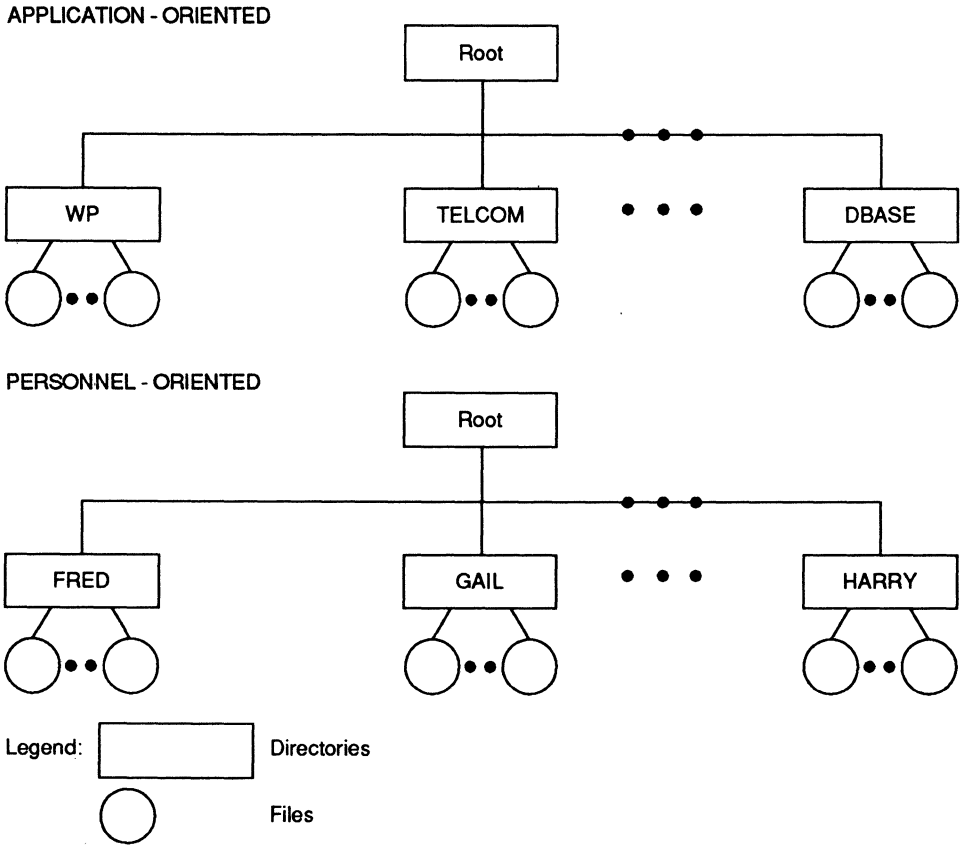


Figure 6.2 Common Directory Structures

under specific subdirectories. Thus, a word processing program and data files created by that program might be placed under a subdirectory named WP, a telecommunications program and its data files under a subdirectory named TELCOM, and so on.

In the lower portion of Figure 6.2, a personnel-oriented directory structure shows a structure common in multiuser systems and networks. This type of directory structure is recommended when several persons share one personal computer. Because the directory structure shown in the lower portion of Figure 6.2 could result in each person maintaining separate copies of application programs, this structure can waste a considerable amount of fixed disk storage. To minimize the potential waste, a combination of an application- and personnel-oriented directory should be considered. Figure 6.3 illustrates an example of this combined directory structure.

In the directory structure illustrated in Figure 6.3, the application program files are placed under specific subdirectories. Other subdirectories with the names or abbreviations of individuals who use specific application programs are created under appropriate application directories. This structure permits individuals to place their data files under specific subdirectories assigned to their use while they can access common application programs placed at specific locations in the directory structure.

Directory and Path Names

Subdirectory names follow the same conventions as standard DOS filenames. That is, they can be up to eight characters in length and can optionally contain a three-character

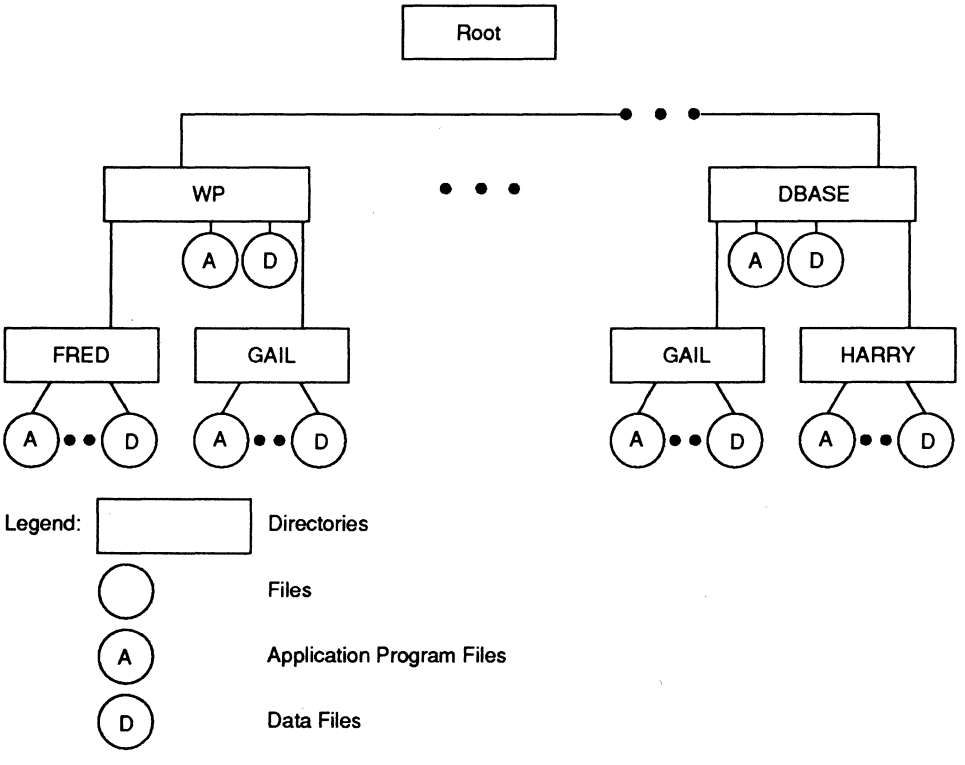


Figure 6.3
Combined Application/Personnel Directory Structure

extension, with a period to separate the directory name from its extension. Thus, although the subdirectory names previously illustrated in Figures 6.1 through 6.3 are perfectly valid, you could add extensions to them to better reflect the contents of a particular subdirectory. In most cases, eight characters are sufficient to define the contents or purpose of a subdirectory, so examples in this chapter use directory names of eight or fewer characters.

Path names are the route the operating system establishes through a hierarchical directory structure to locate a specific file or subdirectory. Path names start at the root and consist of strings of directory names separated from one another by the backslash character. Thus, to access a file named PAY.DAT under the subdirectory FRED in Figure 6.3, your route or path name would become

```
\WP\FRED\PAY.DAT
```

Here, the backslash preceding the WP subdirectory indicates that the route commences at the root directory. If the directory structure illustrated in Figure 6.3 resides on drive C, the complete file specification that defines the location of the file PAY.DAT becomes

```
C:\WP\FRED\PAY.DAT
```

Because each subdirectory has a unique path address, you can assign the same name to files located on different subdirectories. As an example, GAIL could also create a file named PAY.DAT. If GAIL did so, the complete file specification for that file would be

```
C:\WP\GAIL\PAY.DAT
```

With one PAY.DAT file located under Gail's subdirectory and a similarly named file is located under Fred's subdirectory, the operating system considers each as a separate entity. Otherwise, if you attempt to create a file under a subdirectory for which another file with the same name exists, DOS would display an error message. Although the subdirectory structure illustrated in Figure 6.3 appears balanced, in actuality you can nest several levels of subdirectories under one directory and none under another directory or mix a level of nesting under different directories. The only restriction you have to consider is the maximum path length permitted by DOS, which is 63 characters.

Locating DOS

Until now, this chapter has deferred discussion of where the operating system files should be located in a hierarchical directory structure. Although there are essentially an almost infinite number of ways you can structure your disk, most persons prefer to keep their operating system files in one of two popular locations—under the root directory or in a separate subdirectory named DOS. The latter location is where the DOS 4.0 INSTALL program will place the operating system files unless you enter a different subdirectory location.

The advantages associated with locating DOS system files directly under the root are twofold. First, the root directory is the default current directory at which DOS positions you at power-on or if a system reset is performed. Thus, without having to change directories you can immediately enter command keywords. In addition, you do not have to prefix a command with a path to the command file. Second, no matter

where you are located in the hierarchical directory structure, you can simply prefix a command filename with one backslash to inform the operating system that the file is located in the root directory.

The second popular location for DOS files is a separate directory under the root directory that most persons appropriately name DOS. If you use this organization concept, ensure that the file COMMAND.COM, the DOS command interpreter, remains in the root directory. Otherwise, on power-on or a system reset you are placed at the root directory, and DOS will not be able to automatically load COMMAND.COM to interpret subsequent commands entered from the keyboard.

Two other files that must be placed in the root directory are IBMBIO.COM and IBMDOS.COM, because both of these files must be loaded at power-on or system reset to successfully operate your computer. If you want to place COMMAND.COM in the DOS directory, you should place the statement PATH C:\DOS in an AUTOEXEC.BAT file in the root directory. Doing so means that AUTOEXEC.BAT is executed each time your computer is powered-on or a system reset operation is performed, causing the operating system to automatically search the directory named DOS under the root directory for commands or batch files that are not found by a search of the current directory. Then, if you switch to a directory other than DOS and enter the command FORMAT, as an example, the operating system first searches the current directory for the file FORMAT.COM. If it does not find that file in the current directory the operating system then searches the DOS directory for that file. Additional information concerning the use of the PATH command occurs later in this chapter.

A separate directory for DOS command files is primarily created by users who structure their directories. In placing DOS command files in a subdirectory named DOS under the root directory, you must remember to prefix external command keywords with the path \DOS\, with the second backslash followed by the command keyword whenever you are located outside of the DOS subdirectory. As an alternative to prefixing commands with the path \DOS\, you can include the statement PATH C:\DOS in the AUTOEXEC.BAT file, or you can enter this PATH command from the keyboard. Either method causes DOS to search the directory \DOS on drive C after it searches the current directory for commands or batch files.

Directory-Related Commands

DOS includes several commands that can be used to perform such functions as the creation and deletion of subdirectories, as well as to move from one directory to another. To review the operation and use of these commands, follow this section's example of creating a predefined directory structure on a fixed disk and copying several files from different diskettes to different subdirectories on the fixed disk.

Suppose you will be using your personal computer to perform word processing, form design, graphing, database functions, and communications operations, as well as various types of budgeting and financial analysis calculations with a spreadsheet program. In addition, assume the operating system files will be located in a subdirectory named DOS under the root directory. To store your application programs on a fixed disk using separate directories for each category of software, a fixed disk organization that could

satisfy your requirement would be similar to that illustrated in Figure 6.4. To understand the creation of nested subdirectories, assume that two persons would share the use of the form design program. For the two persons, FRED and GAIL, the desired hierarchical directory structure includes subdirectories labeled with their names under the FORMTOOL directory.

Now that the desired directory structure is in place, examine the use of DOS commands to implement this structure. Table 6.1 lists nine commonly used DOS directory-related command prompt commands covered in the remainder of this chapter. The Type column in that table indicates whether the command is internal (I) or external (E). As previously explained in Chapter 5, internal commands are memory resident when DOS

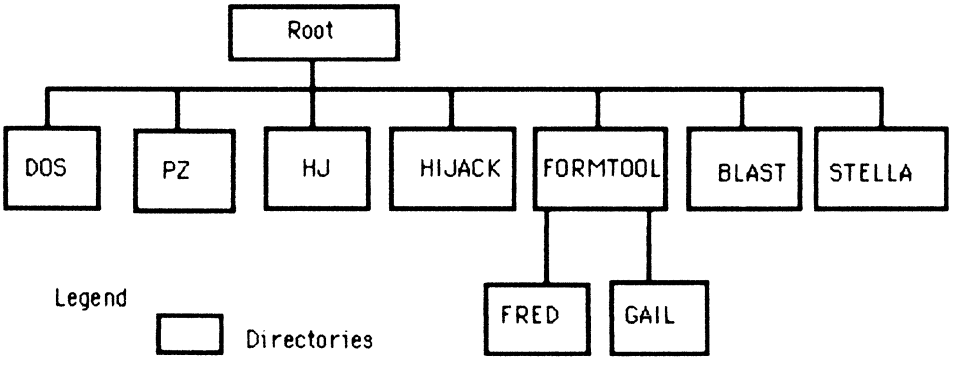


Figure 6.4
Desired Hierarchical Directory Structure

Table 6.1
Commonly Used DOS Directory-Related Commands

Command	Type	Activity Performed
APPEND	I/E	Locates files outside of the current directory with extensions other than .COM, .EXE, or .BAT.
CD or CHDIR	I	Displays or changes the current directory.
JOIN	E	Connects a drive to a directory on another drive, resulting in a single directory structure from two separate directories.
MD or MKDIR	I	Creates a subdirectory.
PATH	I	Establishes one or more routes to different subdirectories DOS will search if a command or batch file is not found in the current directory.
RD or RMDIR	I	Removes a subdirectory from a disk.
SUBSET	E	Assigns a drive identifier to another drive or to a path on a drive.
TREE	E	Displays the directory paths on a drive and optionally lists the files in each directory.
XCOPY	E	Selectively copies groups of files that can include lower-level subdirectories.

is initialized, whereas the code to perform external commands is stored in command files that must be loaded into memory.

In reviewing the operation and use of directory-related commands, assume that your fixed disk was just formatted by the DOS 4.0 INSTALL program and the operating system files were transferred to the subdirectory DOS that is automatically created by that program. Using the directory command (DIR), your directory listing would be similar to that illustrated in Figure 6.5. Note that the DOS 4.0 INSTALL program automatically places the COMMAND.COM file, as well as the AUTOEXEC.BAT file, under the root directory.

MKDIR (Internal)

The MKDIR (make directory) command creates a new directory. The format of this command is

$$\left\{\begin{matrix}\texttt{MD}\\ \texttt{MKDIR}\end{matrix}\right\}\ [d:]path$$

As do several other directory-related commands, this command has an abbreviation. Thus, you can enter either MD or MKDIR for the command name. As an internal command, it does not have to be prefixed with the drive designator and path to a command file, because the command is memory resident when DOS is loaded. With this knowledge, you can use the MKDIR command to create the remainder of the hierarchical directory structure illustrated in Figure 6.4.

To create the six remaining subdirectories that are directly under the root directory, you can simply type MKDIR or MD followed by the appropriate subdirectory name six times. No path is required in the command line, because the current directory is the root directory and the use of the MKDIR command without a path creates a subdirectory under the current directory.

Figure 6.6 illustrates the commands entered at the `C:>` prompt to create the six directories and the use of the DIR command to verify their presence.

In the top portion of Figure 6.6, the entry of MD and MKDIR was purposely varied to illustrate how both keywords can be used to create subdirectories. Note that when the DIR command displays a directory listing, each subdirectory entry is followed by the word DIR to identify the entry as a subdirectory. Otherwise, the entry would be a file and simply show any file extension in this location in the directory listing. At the bottom of the directory listing note the message `9 File(s)`. Although this message may appear confusing because there are seven subdirectories, DOS treats each sub-

Figure 6.5
Initial Directory Structure

```
 Volume in drive C is DOS400
 Volume Serial Number is 2E61-08E9
 Directory of C:\

COMMAND  COM    37637  06-17-88  12:00p
AUTOEXEC BAT      163  03-25-80   1:45a
DOS           <DIR>      03-25-80   1:45a
        9 File(s)     XXXXXXXX bytes free
```

Figure 6.6
Creating Directories Under the Root Directory

```
C:\>MD PZ

C:\>MKDIR HJ

C:\>MD HIJACK

C:\>MKDIR FORMTOOL

C:\>MD BLAST

C:\>MKDIR STELLA

C:\>DIR

   Volume in drive C is DOS400
   Volume Serial Number is 2E61-08E9
   Directory of C:\

COMMAND  COM    37637  06-17-88  12:00p
AUTOEXEC BAT      163  03-25-80   1:45a
DOS           <DIR>      03-25-80   1:45a
PZ            <DIR>      08-29-88   9:32p
HJ            <DIR>      08-29-88   9:33p
HIJACK        <DIR>      08-29-88   9:34p
FORMTOOL      <DIR>      08-29-88   9:36p
BLAST         <DIR>      08-29-88   9:43p
STELLA        <DIR>      08-29-88   9:52p
        9 File(s)     XXXXXXXX bytes free
```

directory as a file to include naming conventions for the subdirectory name and its optional extension.

Now that four subdirectories exist under the root directory, complete your hierarchical directory structure by creating the subdirectories FRED and GAIL under the FORMTOOL subdirectory. In creating these subdirectories, experiment with two different methods using the MKDIR command. The first method involves issuing additional MKDIR commands while you are at the root directory level. To do so, you include the full path names of the desired subdirectories; otherwise, DOS creates the subdirectories under the root directory at the same level as the previously created subdirectories.

To create the subdirectory FRED you could issue either of the following commands using the command keyword MKDIR or the keyword MD.

```
MKDIR\FORMTOOL\FRED

MKDIR FORMTOOL\FRED
```

Similarly, to create the subdirectory GAIL under the FORMTOOL directory, you could enter either of the following commands using the keyword MKDIR or the keyword MD.

```
MKDIR\FORMTOOL\GAIL

MKDIR FORMTOOL\GAIL
```

Assuming that you enter two appropriate make directory commands to create the subdirectories FRED and GAIL and use the DIR command to verify their creation, what is displayed? If you entered the DIR command, the resulting directory would be very similar to that shown in the lower portion of Figure 6.6. In fact, the only difference would be in the number of bytes free, with the directory listing taken after the FRED and GAIL subdirectories were created showing 24 fewer bytes free. An examination of the directory listing shows that the subdirectories FRED and GAIL are conspicuous by their absence from the listing, yet 24 bytes were used. So, an obvious question is, where are the two subdirectories you just created?

If you examine the directory listing in Figure 6.6, which is essentially equivalent to the listing you would see after creating the subdirectories FRED and GAIL, you will note that you are presently located at the root (\) as you perform the directory listing operation. Therefore, the directory command performed as it was designed to operate, only listing those subdirectories and files, if any, located *directly* under the root directory.

To obtain a listing of the subdirectories under the FORMTOOL directory, you must either change directories (which is the next major topic) or indicate to DOS that you want to list the directory of FORMTOOL by specifying the path from the root directory to that subdirectory as shown in Figure 6.7.

Note that the directory command used in Figure 6.7 includes the path \FORMTOOL, informing DOS to list a directory of the contents of the FORMTOOL subdirectory. The directory listing shows four DIR entries, a dot (.), double dot (..), FRED, and GAIL. The single dot designates the current or working directory and is the directory where the system is currently located. The double dot designates the directory immediately above the current directory, also known as the *parent directory.* Both dot entries, as well as subdirectories, are treated as files; hence the entry `4 File(s)` in the last line of the display results from the use of the DIR command.

To solve the mystery of the use of 24 bytes of storage when the subdirectories FRED and GAIL were created, now list the FRED subdirectory. This is illustrated in Figure 6.8. Note that the FRED directory contains two files—dot and double dot. Thus, the creation of a directory entry uses 12 bytes of storage for the two files associated with the directory and their creation date.

DOS Shell Create Directory Operations

When you operate within the DOS Shell, directory-related operations are performed in a manner similar to the file related operations covered in Chapter 5. That is, you must

Figure 6.7
Listing the Contents of the FORMTOOL Subdirectory

```
C:\>DIR \FORMTOOL

   Volume in drive C is DOS400
   Volume Serial Number is 2E61-08E9
   Directory of C:\FORMTOOL

.            <DIR>       08-29-88  9:36p
..           <DIR>       08-29-88  9:36p
FRED         <DIR>       09-03-88 10:36a
GAIL         <DIR>       09-03-88 10:36a
         4 File(s)     XXXXXXXX bytes free

C:\>
```

Figure 6.8
Examining the FRED Subdirectory

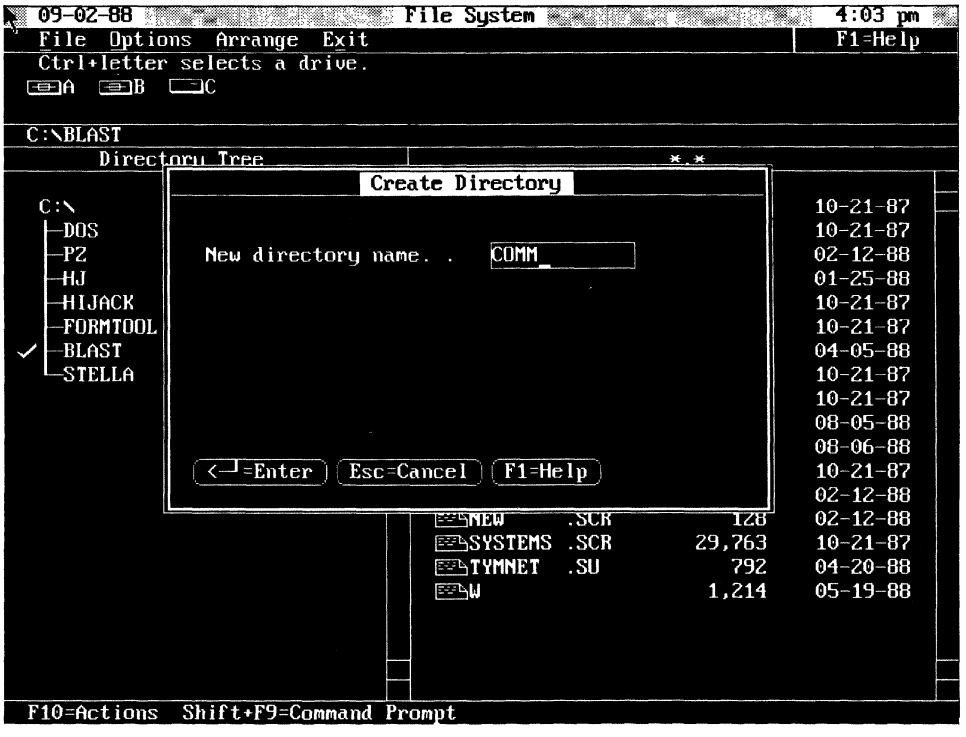
```
C:\>DIR \FORMTOOL\FRED

 Volume in drive C is DOS400
 Volume Serial Number is 2E61-08E9
 Directory of C:\FORMTOOL\FRED

.            <DIR>       09-03-88  10:36a
..           <DIR>       09-03-88  10:36a
       2 File(s)     XXXXXXXX bytes free

C:\>
```

Figure 6.9
DOS 4.0 Create Directory Pop-Up Box



first select a directory or directory location, then select the directory-related operation after you access the File pull-down menu from the File System screen.

The Create Directory operation under the DOS Shell requires you to select the higher-level directory under which you wish to create a new directory. Thus, if your disk was just formatted you would only have the root directory denoted by the backslash (\) symbol under the Directory Tree to select.

Figure 6.9 illustrates the Create Directory pop-up box displayed after you select the `Create directory` option from the File pull-down menu. This menu was previously illustrated in Figure 5.35. Note the check mark to the left of the BLAST directory in the Directory Tree, which indicates that this directory was previously selected. Also

note that COMM has been entered in the New directory name box. Once you enter a new directory name and press the **Enter** key, the new directory is created under the selected directory. Figure 6.10 illustrates the result of the previous Create Directory operation. Note that COMM is now a subdirectory under BLAST and that BLAST is still the selected subdirectory.

Before you examine an alternate method that can be used to create the subdirectories under FORMTOOL, take a moment to review the operation and utilization of the CHDIR command.

CHDIR (Internal)

The CHDIR (change directory) command alters the current or working directory level. The format of this command is

$$\left\{\begin{matrix}\texttt{CD}\\ \texttt{CHDIR}\end{matrix}\right\}\ [d:]path$$

Similar to the Make Directory command, you can use one of two keywords in the Change Directory command line—CD or CHDIR. Using the Change Directory command, you can easily reposition your location to any existing subdirectory, as long as you remember the hierarchical directory structure you created. If you forget your directory

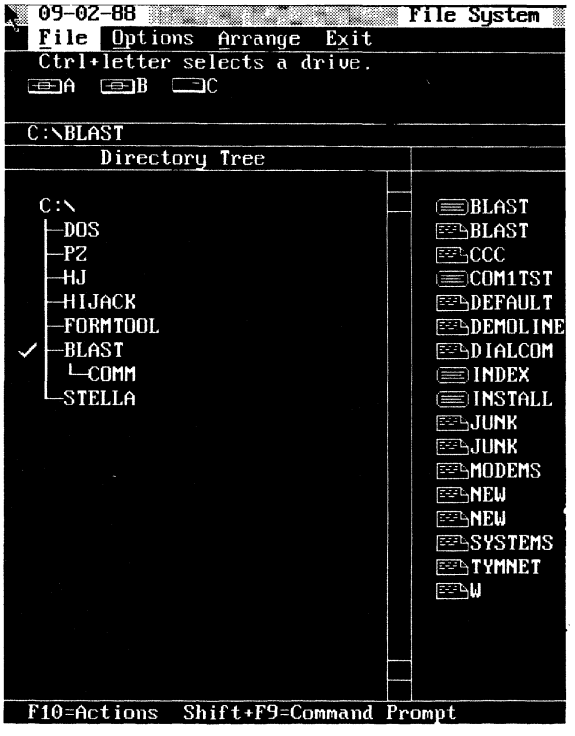


Figure 6.10
Results of the Create Directory Operation

structure, you can use the TREE command to obtain a report. This command is covered later in this chapter.

To move down the hierarchical directory structure to FORMTOOL from the root directory, you would enter the command

```
CHDIR\FORMTOOL
```

Once you have moved the current directory to FORMTOOL, you can issue the DIR command without specifying the path \FORMTOOL and obtain the same result as if you had specified the path when you were located in the root directory. Figure 6.11 shows the alternate method of obtaining a directory listing of a subdirectory without specifying a path in the DIR command.

Again note in Figure 6.11 that the use of the DIR command provides a listing limited to the contents of the current directory. Thus, any files and subdirectories that exist on other directories on the disk are not displayed. After a while this concept will become more obvious, because by noting the message `Directory of...` you will be reminded by DOS that the directory listing is of the subdirectory FORMTOOL.

If you did not previously create the subdirectories FRED and GAIL, you can do so without specifying a path in the MKDIR command, because you previously used the CD command to change the current directory to FORMTOOL. Thus, entering the commands

```
A>MKDIR FRED

A>MKDIR GAIL
```

would create the desired subdirectories under FORMTOOL, without requiring you to specify a path.

To move back to the root directory, simply enter the command

```
CHDIR\
```

Here the backslash as the last character informs DOS to make the root directory the current directory. As an alternative, you can use the double dot (..) designator in a CHDIR command as follows:

Figure 6.11 Obtaining a Directory Listing of a Subdirectory Without Using a Path in the DIR Command

```
C:\>CD\FORMTOOL

C:\FORMTOOL>DIR

   Volume in drive C is DOS400
   Volume Serial Number is 2E61-08E9
   Directory of C:\FORMTOOL

.            <DIR>      08-29-88   9:36p
..           <DIR>      08-29-88   9:36p
FRED         <DIR>      09-03-88  10:36a
GAIL         <DIR>      09-03-88  10:36a
         4 File(s)     XXXXXXXX bytes free

C:\FORMTOOL>
```

Table 6.2 Directory Relationships

Directory	Path	Hierarchical Level
Root	\	root
DOS	\DOS	level 1
PZ	\PZ	level 1
HJ	\HJ	level 1
HIJACK	\HIJACK	level 1
FORMTOOL	\FORMTOOL	level 1
FRED	\FORMTOOL\FRED	level 2
GAIL	\FORMTOOL\GAIL	level 2
BLAST	\BLAST	level 1
STELLA	\STELLA	level 1

```
CHDIR..
```

Because the double dot designates the directory immediately above the current directory and you were located at FORMTOOL, using CHDIR.. moves you back to the root directory level.

Now that you have created the subdirectory structure previously illustrated in Figure 6.4, review that structure with respect to the path and hierarchical level of each directory. Table 6.2 summarizes the relationship between directories to include the path from the root directory and its hierarchical level.

With the DOS Shell you can simply place the selection cursor over the desired directory in the Directory Tree and press the **Enter** key to view the contents of the directory. When you do so, DOS automatically changes directories, which is the reason why there is no "Change Directory" option in the File pull-down menu.

File Operations

When using a hierarchical directory structure, you must include the drive identifier and path to a file unless the file is located on the default drive in the current directory. In such situations, the optional drive identifier and path name are not required, because the default drive and current directory provide DOS with the correct location.

Directory Structure Example

To understand the use of drive identifiers and path names in a hierarchical directory structure, assume you want to load a file named PAY.DAT on a diskette in drive A onto the subdirectory named FRED on drive C. You could use the full DOS command to include all drive identifiers:

COPY A:PAY.DAT C:\FORMTOOL\FRED

If you previously changed the current directory on drive C to FRED by entering CD\FORMTOOL\FRED, you could reenter the previous COPY command as follows:

COPY A:PAY.DAT C:

To verify that the file PAY.DAT was transferred to the subdirectory FRED, you can use the DIR command:

DIR \FORMTOOL\FRED

or, if you changed the current directory to FRED, you could simply enter the command **DIR.**

Figure 6.12 illustrates the use of the "long" DIR command, whose path begins at the root directory to obtain a directory listing of the FRED subdirectory.

In the event FRED is fired and a new employee named ALVIN is assigned to use the form creation program, you can investigate additional directory related operations required to rename a directory. Unfortunately, there is no DOS command under the DOS command prompt mode of operation that renames directories, forcing you to remove the directory named FRED. To do this, you use the Remove Directory command, whose operation and utilization will be covered after you examine how to rename a directory under the DOS Shell.

DOS Shell Rename Directory Operation

Unlike the DOS command prompt mode of operation, which does not provide an easy method to rename a directory, the DOS Shell renames directories simply. The key to renaming a directory is the Rename option in the File pull-down menu on the File System screen.

Similar to file-related operations, you must first select the directory to rename. Use the **Tab** key and the **Up** and **Down arrow** keys to position the selection cursor on the directory to be renamed. Then, press the **Space bar** to select the directory; a check mark is positioned to the left of the selected directory. Now you can press the **F10** key or the **Tab** key to position the selection bar on the File item on the action bar. Then, pressing **Down arrow** displays the File pull-down menu.

Once you select the Rename option, the Rename Directory pop-up box displays over the File System screen, as illustrated in Figure 6.13. Note that the pop-up displays the name of the selected or current directory and positions the cursor at the first position in the New name box. After you enter the new directory name and press the **Enter** key, you see the result of the rename operation, with the new directory name replacing the current name in the Directory Tree area.

Figure 6.12
Verifying the COPY Operation

```
C:\>DIR \FORMTOOL\FRED

 Volume in drive C is DOS400
 Volume Serial Number is 2E61-08E9
 Directory of C:\FORMTOOL\FRED

.            <DIR>        09-03-88  10:36a
..           <DIR>        09-03-88  10:36a
PAY      DAT         54   09-03-88  11:36a
       3 File(s)     XXXXXXXX bytes free

C:\>
```

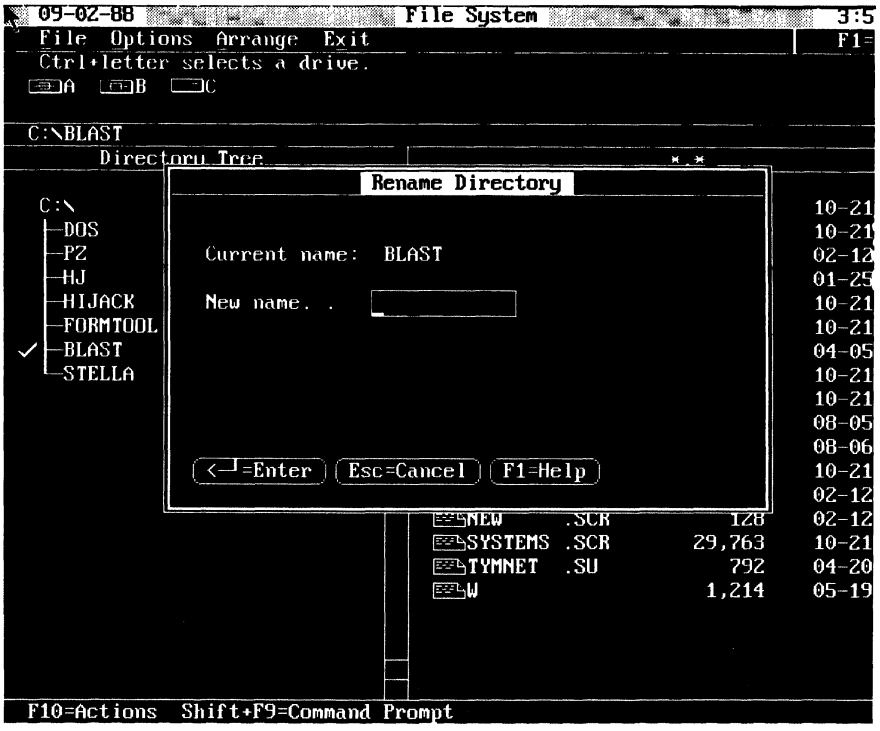


Figure 6.13
DOS 4.0 Rename Directory Pop-Up Box

RMDIR (Remove Directory) (Internal)

The RMDIR (remove directory) command removes a previously created subdirectory. Similar to the directory-related commands described earlier, you can invoke this command using either of two keywords—RMDIR or RD. The format of this command is

$$\left\{\begin{matrix}\text{RD}\\ \text{RMDIR}\end{matrix}\right\} \; [d:]path$$

To successfully use this command, first you must remove all files (except for the dot (.) and double dot (..) entries, which as you recall represent directory levels) from the directory. Then, you must ensure that the current directory is positioned *above* the directory you are removing.

Assuming that the PAY.DAT file under the FRED directory can be used by ALVIN, try creating a subdirectory using the name ALVIN and copying that file to it. Then you can use the DOS ERASE command to remove the file prior to eliminating the subdirectory named FRED.

Assuming the current directory is the root directory, you could use the MD command to create the subdirectory ALVIN under the FORMTOOL directory, as shown here.

C>MD\FORMTOOL\ALVIN

Now, to copy the file PAY.DAT from the subdirectory FRED to the subdirectory ALVIN enter the command

COPY \FORMTOOL\FRED\PAY.DAT \FORMTOOL\ALVIN

Now that the data file has been copied, use the RMDIR command to attempt to remove the subdirectory FRED. After you enter **RMDIR FRED** or **RD FRED**, note the resulting error message. The error occurs because the file PAY.DAT has not been removed from the subdirectory you are trying to eliminate.

```
C>RMDIR \FORMTOOL\FRED

Invalid path, not directory,
or directory not empty
```

Now use the DOS ERASE command to remove the file PAY.DAT, as indicated next.

ERASE \FORMTOOL\FRED\PAY.DAT

Note that if you do not specify a path name, the ERASE command searches the root directory that is the current directory for the file PAY.DAT, which, in this case, is not your intention.

Now that the only file in the FRED subdirectory has been removed, you can remove the FRED subdirectory.

RMDIR\FORMTOOL\FRED

DOS Shell Delete Directory Operation

Similar to the command prompt mode of operation, the DOS Shell requires you to remove all files from a directory prior to deleting the directory.

To delete a directory, you first select it from the Directory Tree area in the File System screen. Assuming that you previously deleted all files in that directory, you can successfully select the `Delete` option from the File pull-down menu. Once this action is accomplished, the Delete Directory pop-up box is displayed. This pop-up box, illustrated in Figure 6.14, displays the current directory that will be deleted and two options you can select. As the figure indicates, the default option is `Do not delete this directory`, forcing you to move the selection cursor to option 2 prior to pressing **Enter**. If you forgot to remove one or more files on the directory to be deleted, the message `Access denied` is displayed.

If you are operating in the DOS command prompt mode, after establishing and deleting directories, as well as copying and erasing files, you might become confused concerning the directory structure you have. Fortunately, DOS includes the TREE command that you can use to generate a report of the structure of a disk.

TREE (External)

The TREE command displays a report of the directory structure of a diskette or hard disk. This report can be limited to the directory paths or can include a listing of all files stored on the disk. The format of this command is

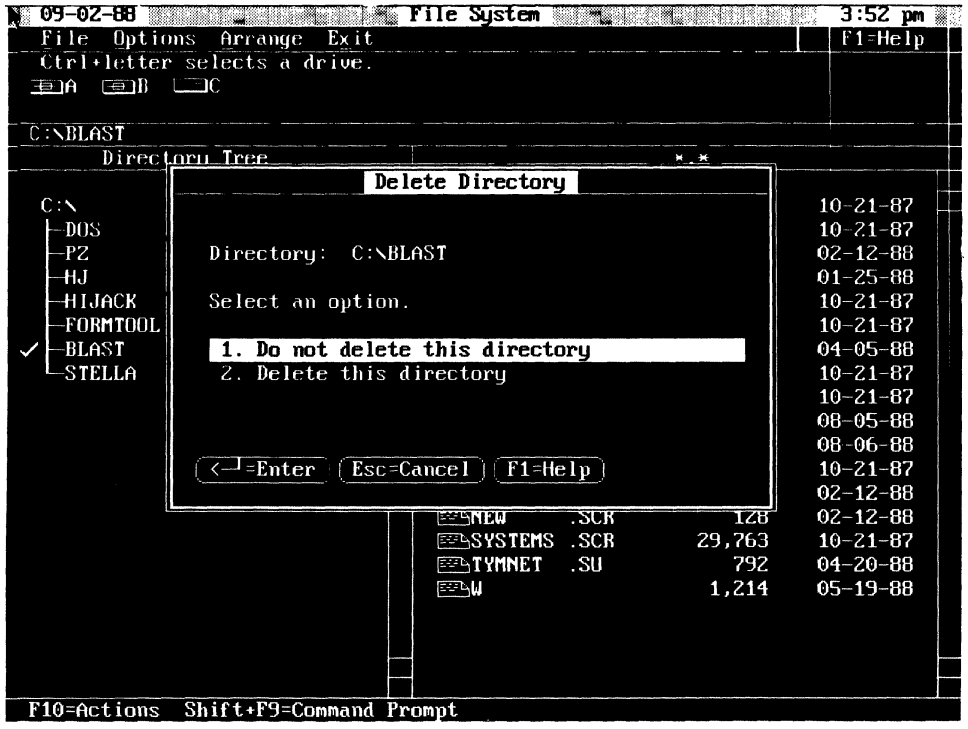


Figure 6.14
DOS 4.0 Delete Directory Pop-Up Box

`[d:][path] TREE [d:][/F]`

Similar to all DOS external commands, the drive identifier and path preceding the command name identify the location of the command file. The drive identifier following the command keyword can specify the drive whose directory paths you want displayed. If not included in the command line, the default drive is used. Finally, the /F option causes all files in each directory to be listed.

Because an extensive directory structure rapidly scrolls off the screen, there are several methods you can consider to better view the directory. First, you can press **Ctrl+PrtSc** to obtain a hardcopy listing of the directory as it is being displayed. If you just wanted to view the directory structure, you could use the Pause Screen function. As another option, you can perform what is known as a piping operation to the DOS MORE filter to display the results of the TREE command one screen at a time. The use of pipes and filters, as well as additional DOS commands to include the MORE command, is covered in Chapter 9.

Assuming you want a report of the directory structure of the disk you have been manipulating in this chapter, you can enter the following command line:

```
C:\>TREE
```

The report generated by the TREE command is illustrated in Figure 6.15. Note that the display of the execution of the TREE command from the DOS command prompt

Figure 6.15
TREE Command Report

```
C:\>TREE
Directory PATH listing for Volume DOS400
Volume Serial Number is 2E61-08E9
C:.
├───DOS
├───PZ
├───HJ
├───HIJACK
├───FORMTOOL
│   ├───ALVIN
│   └───GAIL
├───BLAST
└───STELLA

C:\>
```

mode of operation produces the same display that you will see in the Directory Tree of the File System screen under the DOS Shell. This is why there is no TREE function included in the DOS Shell.

File Location Commands

To simplify the process of accessing files in a hierarchical directory structure, two file location commands are included in DOS—PATH and APPEND. PATH only finds files that can be executed, such as files with the extension .COM, .EXE, and .BAT. APPEND can be considered to be the inverse of PATH, because the command locates files outside the current directory that have extensions other than those PATH looks for.

Because the PATH command was incorporated into earlier versions of DOS prior to IBM adding the APPEND command, it is appropriate to review the operation and use of each command in the order they first appeared in DOS.

PATH (Set Search Directory) (Internal)

The PATH command responds to the names of one or more directory names you enter by telling DOS to search them for executable files after it examines the current directory. The format of this command is

PATH { ; | [[*d*:]*path*[[;[*d*:]*path*]]] }

If you enter the command PATH without any parameters, the current PATH search settings are displayed. Entering PATH with a semicolon resets the search path to null, the initial DOS 3.3 default. If you are using DOS 4.0, the INSTALL program places the command PATH C:\DOS in the AUTOEXEC.BAT file that INSTALL creates in the root directory.

To examine the use of the PATH command, assume you are using DOS 3.3 and wish to automatically access operating system command files located in a subdirectory named DOS under the root directory on drive C. To inform the operating system to search that drive and subdirectory for executable files not found in the current directory, you enter the command

```
C>PATH C:\DOS
```

Thereafter, each time you enter the name of an external DOS command on a command line, DOS automatically searches the DOS subdirectory if the command file is not found in the current directory. To verify the PATH setting, you can enter the command without parameters as shown here.

```
C>PATH

PATH=C:\DOS
```

You can include a list of drives and path names separated by semicolons as long as the total number of characters in the command does not exceed 128. Thus, if you wanted DOS to search the subdirectory FRED located under the FORMTOOL directory, which, in turn, is located under the root directory on drive C, you could modify the command as follows:

```
C>PATH C:\DOS;C:\FORMTOOL\FRED
```

APPEND (Internal/External)

As previously mentioned, the APPEND command can be considered as performing an operation similarly but inversely to the PATH command. That is, the APPEND command enables DOS to search for files outside of the current directory whose extensions are other than .COM, .EXE, and .BAT. Thus, using this command you can store applications once on a fixed disk and use them without changing to the directory in which they are located.

The first time you use the APPEND command, it functions as an external command and will be loaded from a file into memory. The format for this command when used as an external file is

```
[d:][path]APPEND {d:path[;[d:][path...]]
                 {APPEND[/X][/E]
```

As with all external commands, the drive specifier and path preceding the keyword APPEND are used to specify the location where the command file resides. If the command is used without its optional /X and /E parameters, you can specify the paths to subdirectories DOS should search to access nonexecutable files stored outside the current directory. This will probably be the most popular format of the APPEND command, because it enables you to access overlay files and additional software such as help files that PATH does not provide access to. Very few programs are made up of a single executable program, so you will probably use the PATH command followed by an APPEND command with similar syntax to obtain full access to modern application programs.

Thus, to complement the previously discussed PATH command, you could enter the APPEND command as follows:

```
C>APPEND C:\DOS;C:\FORMTOOL\FRED
```

This APPEND command would cause DOS to automatically search the subdirectories DOS and FRED for nonexecutable files not found in the current directory.

The /X optional parameter causes paths specified by a previous APPEND command to be searched on the occurrence of certain function calls. If you specify the /E parameter, the current APPEND path is stored in the DOS environment and can be changed by the SET command. The DOS environment and several additional commands are discussed in Chapter 8.

Once APPEND has been loaded it functions as an internal command for the duration of the current work session. At this time its format becomes

```
APPEND { d:path[;[d:]path...] }
       { [;]                  }
```

When you use the APPEND command with a semicolon, any previous APPEND list of paths is reset to null, which is its default value when DOS is initialized.

The only action under the DOS Shell that performs a similar function to the APPEND command is `Associate`. This action, which is invoked from the File pull-down menu on the File System screen, enables you to associate filename extensions with a program. Then, whenever you select a file with the previously specified extension and also select the `Open` (start) action, the program associated with the selected file is invoked. This feature enables you to keep data files in separate directories from program files but select a data file to start the associated program.

Utility Commands

This section examines the operation and use of three DOS directory-related utility commands—XCOPY, JOIN, and SUBST. XCOPY is similar to, but much more powerful than, the DOS COPY command. XCOPY helps you to selectively copy groups of files that can include lower-level subdirectories. JOIN enables you to form a single directory structure from two separate directories, whereas SUBST enables you to assign a drive identifier to another drive or to a path on a drive.

XCOPY (External)

The XCOPY command selectively copies groups of files, including lower-level subdirectories. The format of this command is

```
[d:][path]XCOPY { [d:][path]filename[.ext]  } [d:][path][filename[.ext]]parameters
                { [d:]path[filename[.ext]]  }
                { d:[path][filename[.ext]]  }
```

where the optional parameters are

/A to copy only files whose archive bit was set to one without changing the bit setting.

/D to copy only files whose date is the same or later than the date specified by entering the date format in one of the following formats:

/D:*mm-dd-yy*

/D:*dd-mm-yy*

/D:*yy-mm-dd*

where the date format entered is based on the country code you specified when the SELECT or COUNTRY command was used.

/E causes subdirectories to be created on the target drive, even if they are empty after all copying is completed. If /E is not specified, empty subdirectories are not created.

/M copies only those files whose archive bit is set to one, then turns off the archive bit of the source file.

/P copies on a file-by-file basis, prompting you prior to copying each file.

/S copies all files on the source disk regardless of subdirectory position to be copied. This optional parameter does not create an empty subdirectory on the target disk unless the /E parameter is also specified in the command. If the /S parameter is not included, only files with the specified or current directory are copied.

/V causes DOS to verify that data copied to the target disk is recorded properly.

/W causes the command to display the message

```
Press any key to begin copying file(s)
```

This parameter provides you with the opportunity to insert different diskettes before the XCOPY command commences.

To illustrate the use of the XCOPY command, assume the diskette in drive A and your fixed disk in drive C have the tree structures illustrated at the top of Figure 6.16.

By entering the XCOPY command as shown here, the resulting target directory structure illustrated at the bottom of Figure 6.16 is created.

XCOPY A:\ C:\FORMTOOL /S

In this XCOPY command, A:\ informs DOS that the root directory of drive A is the starting point for the command. The identifier and path C:\FORMTOOL specifies the target drive and path XCOPY will copy files to. The /S parameter tells DOS to copy all files below the starting source directory.

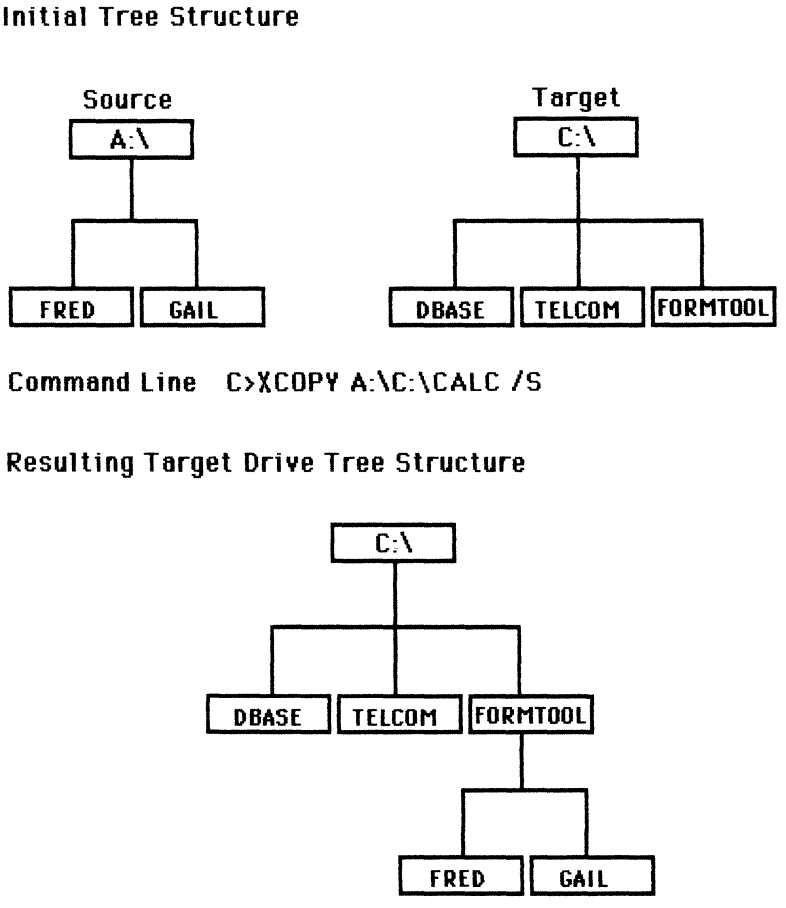
If you prefer to copy specific files, you can add filenames to the preceding example. Assuming you only want to copy files with the extension .ASC, you can enter the following command line.

XCOPY A:*.ASC C:\FORMTOOL /S

If you want to copy all files with the extension .ASC but rename the extension of each file to .DAT, you can enter the following command line:

XCOPY A:*.ASC C:\FORMTOOL*.DAT /S

Figure 6.16 Using the XCOPY Command



JOIN (External)

The JOIN command can connect a drive to a directory on another drive, forming a single directory structure. Through this command, you can let application programs that are diskette resident take advantage of the storage capacity of fixed and virtual disks. The format of this command is

```
[d:][path]JOIN {[d: d:\directory]}
               {[d: /D]         }
```

As for all external commands, the drive identifier and path preceding the command identify the location where the JOIN command file resides. If you enter the command without any parameters, the currently joined drives and directories, if any, are displayed.

The drive identifier immediately after JOIN denotes the drive to be connected or disconnected to or from a directory on another drive. If the drive identifier is followed by a second drive identifier and directory path, the first drive is joined to the second

drive under the specified directory. Note that the specified directory must contain no files other than dot (.) and double dot (..) if the directory has been created already. If it does not exist, DOS creates the specified directory. The resulting operation of the JOIN command is illustrated in Figure 6.17. If the parameter /D follows the first drive identifier, the specified drive is disconnected from any previously JOINed directory.

Figure 6.18 illustrates the operation and verification of a JOIN command. Note that the contents of the diskette in drive A are joined to drive C as a directory named DISKETTE. Then, the DIR command verifies the operation of the JOIN command.

SUBST (Substitute) (External)

The SUBST (Substitute) command assigns a drive to a path on another drive. The primary use for this command is to permit application programs that do not recognize paths to do so by making a subdirectory masquerade as a disk drive. That is, SUBST enables you to assign a drive identifier to an existing disk drive and subdirectory, permitting data routed to the assigned disk drive and subdirectory in actuality to flow to the assigned drive.

The format of the SUBST command is

$$[d:][path]\text{SUBST}\ \left\{\begin{matrix}[d:\ d:path]\\ [d:\ /\text{D}]\end{matrix}\right\}$$

Figure 6.17 Using the JOIN Command

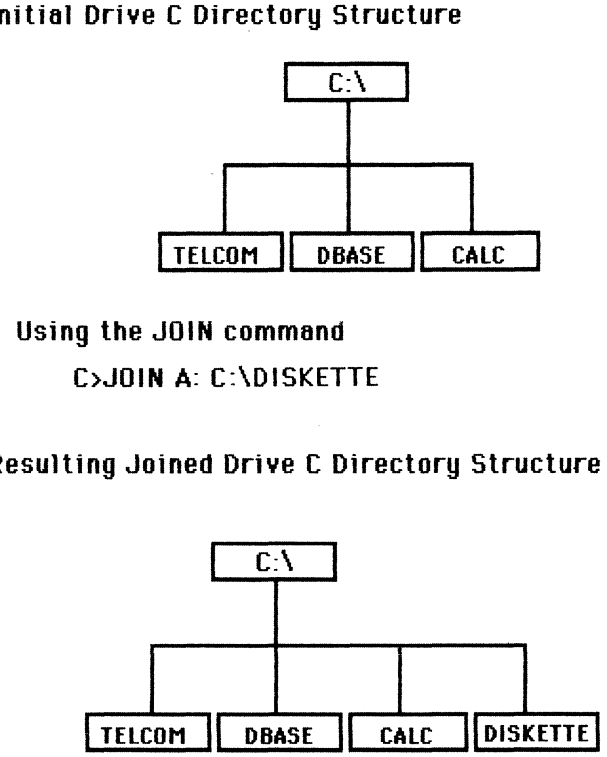


Figure 6.18
Verifying the Use of the JOIN Command

```
C:\>JOIN A: C:\DISKETTE

C:\>DIR

   Volume in drive C is DOS400
   Volume Serial Number is 2E61-08E9
   Directory of C:\

COMMAND  COM    37637  06-17-88  12:00p
AUTOEXEC BAT      163  03-25-80   1:45a
DISKETTE      <DIR>      09-03-88   1:27p
DOS           <DIR>      03-25-80   1:45a
PZ            <DIR>      08-29-88   9:32p
HJ            <DIR>      08-29-88   9:33p
HIJACK        <DIR>      08-29-88   9:34p
FORMTOOL      <DIR>      08-29-88   9:36p
BLAST         <DIR>      08-29-88   9:43p
STELLA        <DIR>      08-29-88   9:52p
        10 File(s)     XXXXXXXX bytes free

C:\>
```

The first drive identifier after SUBST denotes the drive to be substituted for or have its association deleted from another drive or path. This drive identifier can be a physical or a logical drive. If the drive identifier is higher than E, which is the highest DOS normally permits, you must include a LASTDRIVE assignment in your CONFIG.SYS file.

The second drive identifier and path following the SUBST keyword references the drive and subdirectory you want to operate as if it was an independent disk drive. If you follow the first drive identifier with the parameter /D, any previously set substitution is deleted, whereas simply specifying the keyword without any parameters displays the currently assigned substitutions.

As an example of the use of the SUBST command, assume drive C has the directory such that the path C:\FORMTOOL\GAIL exists on that drive. Issuing the command

SUBST E: C:\FORMTOOL\GAIL

provides you with two ways to access the subdirectory GAIL on drive C as a result of the substitution. You can still use the physical drive and path name, or you could use the disk drive identifier E. Once the substitution is accomplished disk drive E, in effect, becomes an abbreviated method of referencing the path \FORMTOOL\GAIL on drive C.

7 EDLIN

This chapter examines the operation and use of the text line editor included on the DOS Operating diskette.

Uses of EDLIN

Through EDLIN you can create, modify, and alter ASCII programs and text files. Although you can purchase a more sophisticated text editor or use a word processor that supports ASCII file I/O operations, EDLIN is normally sufficient in capability to use for the creation of short batch files, including the AUTOEXEC.BAT and CONFIG.SYS files.

To create very short batch files, many users prefer to enter the DOS COPY command, copying input from the keyboard (CON) directly to a batch file. Then, pressing the Ctrl+Z multikey combination writes an end-of-file mark, terminating keyboard input to the file. Whereas this method of ASCII file creation is normally acceptable for entering a few lines of data, it provides no editing capability after a line is terminated by pressing the Enter key. Thus, although EDLIN may pale in comparison to a word processor, its inclusion on the DOS diskette provides a line text editing capability at no extra cost that is sufficient for creating, modifying, and displaying short ASCII files.

Operation of EDLIN

The EDLIN text editor is an executable program file that is normally located in the same directory in which you place the contents of your DOS diskette. If you use the INSTALL program under DOS 4.0, you probably placed EDLIN in the DOS subdirectory under the root directory and have a PATH C:\DOS statement in an AUTOEXEC.BAT file that results in the operating system searching that directory if it cannot find the requested .COM, .EXE, or .BAT file in the current directory. Due to this, under DOS 4.0 if you have a fixed disk you can invoke EDLIN using the following command format:

```
EDLIN [d:][path]filename
```

If you are operating under the DOS Shell, you can execute EDLIN from the File System screen. Simply press the **Tab** key to locate the selection cursor in the Directory Tree area. Then you can move the selection cursor with the **Up** or **Down arrow** keys to the directory where EDLIN is located and press the **Enter** key to display the file

EDLIN.COM in the filename area. Then you can press **Tab** and use the **Up** and **Down arrow** keys to position the selection cursor over the EDLIN.COM file and press the **Space bar** to select that file. Finally, you can press the **F10** key to select the `File` item in the action bar, use the **Down arrow** key to pull down the File menu, and press **Enter**; the selection cursor is positioned over the `Open (start)` option, which is the first action in that menu. This displays the Open File pop-up box with EDLIN.COM as the starting program entry.

If you have a diskette-based system or you are using DOS 3.3 without an appropriate PATH statement, you should prefix the keyword EDLIN with the drive and/or path to the file if it is located on a different disk and/or a different directory.

If the filename represents a new file you are creating, the message `New file` is displayed, followed by an asterisk (`*`) on a new line, which is the EDLIN prompt. The following example illustrates how to invoke EDLIN to create an ASCII file named TEST.BAT on drive A.

```
C:\>EDLIN A:TEST.BAT
New file
*
```

If you are using EDLIN to edit a previously created file, the message `End of input file` is displayed instead of `New file`. When EDLIN loads a previously created file, it uses a 64K byte area of storage. If the text editor cannot load the entire file into three quarters of its reserved storage area, it displays its prompt in place of the `End of input file` message. At that time you can edit the portion of the file that is memory resident. Then you can write a portion of the edited lines to disk and load unedited lines from disk to continue the editing process.

Using Line Numbers

During the editing process, EDLIN dynamically generates and displays line numbers. These line numbers function as a reference to the lines in the file; however, they are not included in the actual file. Thus, if you are editing a program file that uses line numbers, such as a BASIC interpreter program, the display of program lines may appear awkward at first, because you see two line numbers on each line. The first line number is generated and displayed by EDLIN as a reference line for editing purposes but is not included in the file when it is saved. The following example illustrates the use of EDLIN to display the first five lines from a BASIC program contained on the file STAT.BAS on drive A. In this example, the EDLIN L command is used to list the first five lines of the STAT.BAS file. A later section in this chapter examines the format and use of each EDLIN command in detail.

```
C:\>EDLIN A:STAT.BAS
End of input file
*1,5L
       1:*10 CLS
       2: 20 LOCATE 5,10
       3: 30 PRINT "STATISTICAL ANALYSIS"
```

```
        4: 40 PRINT "VERSION 4.0"
        5: 50 PRINT
*
```

The asterisk displayed following the colon in EDLIN line 1, which is line 10 of the STAT.BAS program, denotes the position of the current line in the file. The current line can be considered as a reference or baseline against which commands operate when you make changes to a file.

Functional Capacity

As a line editor, EDLIN operates on one line at a time. This means that any file used by EDLIN is actually treated as a series of lines, with each line ending with a carriage return.

A line in EDLIN can be up to 253 characters in length, with the data between two carriage returns treated as a line entry. Lines in memory are automatically numbered beginning at 1 when you use EDLIN. These numbers, as previously mentioned, are not a part of the file. Thus, the EDLIN line numbers are not counted in the 253-character line limit.

Using EDLIN you can add, delete, copy, and move lines. When you perform any of these operations, EDLIN automatically adjusts its line number referencing scheme to correspond to the operational result of line manipulation commands.

To ensure that users do not inadvertently change the contents of a file and lose the ability to reconstruct their data, EDLIN automatically creates a backup file if you modify any data on the file being edited. The original file is renamed by EDLIN with an extension of .BAK, whereas the new file has the filename and extension specified in the EDLIN command. If you should later desire to use EDLIN to edit a backup file, you must first rename the file with another extension or use the COPY command to create another version of the file with a different extension. Once you have renamed the backup file with a new extension or copied it using an extension different than .BAK, you can start EDLIN and specify the new filename.

EDLIN Commands

There are 14 commands you can use with EDLIN. Table 7.1 summarizes each command, including its format and operational result.

I (INSERT LINES) Command

The INSERT LINES command adds lines to a file under EDLIN. If you are creating a new file, this command must be used prior to inserting text. If you are editing a previously created file, use this command to add lines at a specified position, at the current line, or at the end of the file. Prior to examining the insertion of lines in an existing file, you will find it helpful to examine the use of the INSERT LINES command used in creating a new file.

New File Use

When you invoke EDLIN specifying the name of a new file, the prompt `New file` followed by an asterisk on a separate line is displayed. At this time, you will note that

Table 7.1
EDLIN Command Summary

Command	Operation	EDLIN Format
APPEND	Adds the specified number of lines from disk to the file in memory.	[n]A
COPY	Copies the lines in the range defined by the first two line numbers to the third line number. The count specifies the number of times to repeat the text.	line,line,line,countC
DELETE	Deletes the lines in the range; the line after the deleted range becomes the current line. The current line is deleted if a period is entered, while the last line is deleted if a # is specified.	{ [line][,line] / [.] / [#] } D
EDIT	Displays the specified line if a line number is entered or the current line if a period is entered.	{ [.] / [line] }
END EDIT	Ends EDLIN and saves the edited file.	E
INSERT	Inserts lines of text immediately before the specified line if a line number is entered, the current line if a period is entered, or after the last line if a # is specified.	{ [.] / [line] / [#] } I
LIST	Displays the specified line or range of lines, leaving the current line position unchanged.	[line][,line]L
MOVE	Moves the lines in the specified range to the line number specified by the third line parameter.	line,line,lineM
PAGE	Lists the specified line or range of lines, changing the current line to the last line displayed by the command.	[line][,line]P
QUIT EDIT	Quits the editing session without saving changes.	Q
REPLACE TEXT	Replaces all occurrences of the first string in the specified line or range of lines with the second string. The ? requests a prompt OK? after each display of a changed line.	[line][,line][?]Rstring [F6string]
SEARCH TEXT	Searches the specified line or range of lines to find the specified string. The ? requests a prompt OK? after each display of a line containing the specified string.	[line][,line][?]Sstring
TRANSFER	Causes the contents of the specified file to be merged ahead of the line in the file you are currently editing.	[line]T[*d*:]filename
WRITE LINES	Writes the specified lines to disk from the lines in memory being edited	[n]W

the asterisk is not the current line pointer, because no line numbers are displayed. This is indicated by the following example.

```
C:\>EDLIN A:TEST.DAT
New file
*
```

No line numbers are displayed because at this time the file does not contain any lines of data. Thus, the asterisk is informing you that EDLIN is expecting a command.

To enter data, you switch EDLIN to its insert mode. This is accomplished by simply typing **I** for Insert and pressing the **Enter** key, as shown.

```
New file
*I
       1:*_
```

Note that EDLIN has now displayed a line number followed by a colon and asterisk, with the asterisk denoting the current line. The underline following the asterisk indicates the position of the cursor. At this point, EDLIN is in its insert mode. Now you can enter data on line 1. When you press **Enter**, assuming you typed **Line one**, you see

```
New file
*I
       1:*Line one
       2:*_
```

At this point, line 2 has become the current line, which is indicated by the last line number marked with an asterisk. Suppose you entered four lines of data, as indicated.

```
New file
*I
       1:*Line one
       2:*Line two
       3:*Line three
       4:*Line four
       5:*_
```

If you want to terminate input and return to EDLIN's command mode, you must press the Ctrl+Break or Ctrl-Z multikey combination. Doing so displays ^C on your screen and a new asterisk in column 1 of a new line, followed by the cursor. The asterisk informs you that EDLIN is ready for the entry of a new command. This sequence is

```
New file
*I
       1:*Line one
       2:*Line two
       3:*Line three
       4:*Line four
       5:*^
*_
```

What happens if you again place EDLIN in its insert mode? If you enter I after the command prompt asterisk, EDLIN displays the current line followed by a colon and asterisk, with the cursor positioned after the current line indicator (asterisk). This is illustrated here.

```
*I
       5:*_
```

Line Insertion

You can insert new lines into a file by prefixing the I command with a line number, the decimal point, or the # symbol. Using a line number, such as 3I, causes EDLIN to insert text before line 3. EDLIN displays the line number you used to prefix the I command, followed by a colon and an asterisk. The following example illustrates how you could add an extra line to the new file you are creating in this sample session, positioning the line after line 2.

```
*3I
       3:*Line after line two
       4:*^C
*_
```

To verify the position of the inserted line, you can use the LIST command (L) without any parameters, as shown:

```
*L
       1: Line one
       2: Line two
       3: Line after line two
       4:*Line three
       5: Line four
*_
```

Note that as a result of the insertion process, the lines after line 3 were renumbered. Also note that the L command displays only one asterisk, which denotes the current line. The other methods used to insert lines include prefixing the I command with a decimal point or the # symbol. Prefixing the I command with a decimal point enables you to insert a line of text immediately before the current line. Prefixing the I command with the # symbol makes the insertion after the last line in memory.

L (LIST LINES) Command

You use the LIST LINES command to view a single line or a specified range of lines. Similar to the I command, you can prefix the L command with a period or the # symbol. When you use the period, the current line is displayed, whereas the # symbol causes the line after the last line in memory to be displayed.

The basic format of the L command is

```
[line][,line]L
```

If you omit both line parameters and simply enter L, EDLIN displays up to 24 lines from the file being edited. Up to 11 lines are displayed before the current line and 12 lines after the current line. If 11 lines do not exist before the current line, EDLIN displays extra lines after the current line to make a total of 24 lines being displayed.

To display a range of lines, you enter two line numbers separated by a comma prefixing the letter L. Thus, entering 5,12L displays lines 5 through 12. If you omit the first parameter, such as entering ,12L, the display starts 11 lines before the current line and ends with line 12. Note that the beginning comma is required to indicate that the first line parameter was omitted. If you omit the second parameter, up to 24 lines are displayed, starting with the specified line. When the second parameter is omitted, you can use either of these two formats:

```
lineL
line,L
```

P (PAGE) Command

The PAGE command is very similar to the LIST LINES command. Although both commands can be used to display a line or range of lines, PAGE changes the current line to the last line displayed. Thus, if the current line were line 3 and you entered the command 1,5P, lines 1 through 5 would be displayed and line 5 would become the current line.

EDIT Command

Now that you have entered a few lines of text, take a look at how to edit the data on a line. To do so, try the EDIT command for editing a specific line by entering the line number followed by the Enter key. Similar to the I and L commands, you can enter a period to edit the current line or the # symbol to edit the last line in memory.

As an example of the use of EDIT, try editing line 3 from the previous example. The EDLIN * prompt should be displayed in column one, or you can press the **Ctrl+Break** multikey combination to display that prompt. Assuming the EDLIN * prompt is displayed, the result of entering line 3 for editing is

```
*3
       3:*Line after line two
       3:*_
```

Note that EDLIN displays the requested line for editing as well as making that line the current line. At this time, you can type a completely new line and press **Enter** to store it in memory. If, after you begin editing a line, you change your mind and want to keep its contents as they are, you can cancel editing by pressing **Ctrl+Break** or **Esc**. Doing so leaves the original line unchanged and redisplays the EDLIN command prompt (*) in column one.

When editing a line, you can use the DOS editing keys (F1 through F5), as well as the Ins and Del keys, to make changes to a previously entered line. DOS editing keys are applicable to EDLIN, because the text editor uses an input buffer to hold the contents of an edited line similar to the manner in which a DOS command line is held. This

enables you to operate on the contents of the input buffer. Then, when you are satisfied with the data on the screen that reflects the contents of the input buffer, you can press **Enter** to have the contents of the input buffer replace the line entry position in memory.

In reviewing the use of the DOS function keys and the Ins and Del keys under EDLIN, return to the previously entered editing of line 3. Because the F1 key displays a character from the input buffer each time that key is pressed, pressing F1 five times causes the screen to appear as follows:

```
*3
       3:*Line after line two
       3:*Line _
```

Now you could press the **Ins** key and type **is**. Then, pressing the **F3** key you can copy the rest of the line from the input buffer to your screen resulting in the following display.

```
*3
       3:*Line after line two
       3:*Line is after line two_
```

Now press the **Enter** key to place the revised line into storage or add additional text to the end of the line if you wish.

Similar to its use at the DOS command prompt level, the F2 key copies characters up to a specified character. Thus, if the screen appears as follows:

```
*3
       3:*Line after line two
       3:*_
```

and you press **F2**, type the letter **e**, and press **Enter**, your display appears as follows:

```
*3
       3:*Line after line two
       3:*Line _
```

The remaining function keys for editing—F4 and F5—also function exactly the same as when you use them to edit a DOS command line entry under DOS 3.3. However, they are not applicable for use under DOS 4.0. Press **F4** followed by a single character and **Enter** to delete all characters in the input buffer from the cursor location to the next occurrence of the specified character. Press **F5** to place the currently edited line into the input buffer for additional editing. Although the screen display appears similar to what you would see if you pressed Enter, the line will not be a permanent entry in memory but merely placed in the input buffer. To denote this, when you press F5 the @ character is displayed at the end of the line.

D (DELETE) Lines Command

The DELETE command removes a specific line or a range of lines from the file in memory. Similar to other EDLIN commands that work on a line or range of lines, with D you can enter a period to delete the current line or the # symbol to specify the line

after the last line in memory. Once a line or range of lines is deleted, the line following the line or range of lines becomes the current line. To examine the use of the DELETE command, assume the listing of the current file results in the following display.

```
*L
       1: Line one
       2: Line two
       3: Line three
       4:*Line four
       5: Line five
```

Entering the command D by itself deletes the current line, which is line 4, renumbering line 5 as 4 and making it the current line. The command 2D would delete line 2, renumbering lines 3 through 5 as lines 2, 3, and 4, respectively, with line 3 then becoming the current line. If you entered the command 2,4D, lines 2 through 4 would be deleted, line 5 renumbered as line 2, which becomes the current line.

S (SEARCH TEXT) Command

If you are editing a lengthy file, the SEARCH TEXT command can rapidly locate a specified string. You can enter the SEARCH command with a specific line number, a range of line numbers, or without specifying a line number. Assuming you want to locate the string `four`, review some of the options available in searching a file for this string.

When you use the SEARCH TEXT command, EDLIN examines the specified line or range of lines for the specified string. The first line that contains the specified line is displayed, and the search terminates unless the ? has been included in the command line. Including the ? in the command line causes the prompt `O.K.?` to be generated after each display of a matching string. Then, entering **Y** terminates the search, whereas entering **N** causes EDLIN to search for the next occurrence of the string in a range of lines.

Assume you are working with the following file:

```
*L
       1: Line one
       2: Line two
       3: Line three
       4:*Line four
       5: Line five
```

The result of entering the command Sfive is

```
       5: Line five
*_
```

Now, you could edit that line by typing a period or the number 5 and pressing the **Enter** key.

Entering the command 1,5SLine would produce the following result:

```
*1,5SLine
       1: Line one
*_
```

Entering the letter S by itself results in the search being continued using the previously specified string. The process is shown in this example.

```
*S
       2: Line two
```

If you enter the ? parameter, EDLIN asks you after each display of a matching line whether to continue the search. This is illustrated by the following example:

```
*1,5 ? SLine
       1: Line one
O.K.?N
       2: Line two
O.K.?Y
*_
```

R (REPLACE TEXT) Command

You can use the REPLACE TEXT command to replace one string with a second string on a specified line or within a range of lines. Similar to the SEARCH TEXT command, you can use the ? parameter to have EDLIN ask you whether the replacement should occur for a specific line.

As indicated in Table 7.1, all occurrences of the first string that follow the character R in the specified range of lines will be replaced by the second string. If you omit the second string from the command line, all occurrences of the first string within the specified range of lines will be deleted. Thus, entering

`1,5RLine` **F6** `Position`

where **F6** indicates pressing the F6 key, would cause each occurrence of the string `Line` in lines 1 through 5 in memory to be replaced by the string `Position`. Note that you can either press the **Ctrl+Z** multikey combination or **F6** to indicate the termination of the first string.

When you use the REPLACE TEXT command, unless you include the ? parameter all replacements are displayed on your screen as they occur. This is illustrated by the following example.

```
*,5RLine^ZPosition
       1: Position one
       2: Position two
       3: Position three
       4: Position four
       5: Position five
*
```

If you enter 1,5RLineF6 and press the **Enter** key without entering a second string all occurrences of Line in lines 1 through 5 will be deleted.

C (COPY LINES) Command

The COPY LINES command duplicates one line or a range of lines at another location in memory. By including a count value at the end of a COPY LINES command, you can repeat the copying operation a specified number of times. If you do not specify a count value, a default of 1 is used.

Similar to many other EDLIN commands, you can use the period to specify the current line and the # symbol to specify the line after the last line in memory. Once you enter the COPY LINES command, the lines in the specified range denoted by the first and second line parameters are copied to the line number specified by the third line number, ahead of that line.

As an example of the use of the COPY LINES command, consider the following examples.

```
1,3,12C
```

This causes lines 1 through 3 to be copied ahead of line 12, with line 12 becoming the current line. Similarly,

```
1,3,12,2C
```

copies lines 1 through 3 ahead of line 12, repeating the copying operation twice.

To better visualize the operation of the COPY LINES command, consider its use in Listing 7.1. At the top of that illustration, the LIST LINES command lists the contents of the file in memory. Next, the COPY LINES command copies lines 1 and 2 to the line after the last line in memory. Finally, another LIST LINES command lists the revised contents of memory. Note that lines 1 and 2 are copied to lines 6 and 7 and that line 6 is now the current line.

Listing 7.1 Using the COPY LINES Command

```
*1
       1: Line one
       2: Line two
       3: Line three
       4: Line four
       5:*Line five
*1,2,#c
*1
       1: Line one
       2: Line two
       3: Line three
       4: Line four
       5: Line five
       6:*Line one
```

```
       7: Line two
*
```

M (MOVE LINES) Command

The MOVE LINES command transfers a range of lines from one location in a file to another. The first and second line parameters in the command specify the range to be moved, whereas the third line number specifies the location where the data will be moved. If you do not specify a range, the current line is used as the default.

With the MOVE LINES command you use the + and – characters to specify relative line locations. As an example, you can enter the following command to move data from the current line plus 20 lines to line 100:

,+20,100M

Similar to other EDLIN commands, you can use the # symbol to specify the line after the last line in memory and the period to specify the current line. Thus, you can enter the following command to move data from lines 20 through 50 to behind the last line in memory.

20,50,#M

Listing 7.2 illustrates the use of the MOVE TEXT command. At the top of that illustration, the LIST LINES command is entered to display the contents of memory. Next, a MOVE TEXT command moves lines 6 through 7 to precede line 1. Then another LIST LINES command verifies the relocation of the two lines.

Listing 7.2 Using the MOVE TEXT Command

```
*L
       1: Line one
       2: Line two
       3: Line three
       4: Line four
       5: Line five
       6:*Line one
       7: Line two
*6,7,1M
*L
       1:*Line one
       2: Line two
       3: Line one
       4: Line two
       5: Line three
       6: Line four
       7: Line five
*
```

File Reference Commands

EDLIN contains four commands that operate on files—END EDIT, QUIT EDIT, TRANSFER LINES, and WRITE LINES. This section reviews the operation and utilization of each of these commands.

E (END EDIT) Command

The END EDIT command terminates an EDLIN session and saves your updated file.

Once you enter the command, the edited file is saved by the system writing it to the drive under the file specification established when you invoked EDLIN. If you are editing the file TEST.BAT on drive A, after you enter **E** at the command prompt (*) level EDLIN returns you to the DOS command level denoted by the drive designator followed by the greater than (>) symbol. If you are operating under the DOS Shell, the File System screen will appear once EDLIN is terminated.

Q (QUIT EDIT) Command

In comparison to the END EDIT command, which saves the updated file and returns you to the DOS prompt, the QUIT EDIT command results in the exiting of the editing session without saving changes.

Entering the character Q at the EDLIN command prompt (*) causes the text editor to terminate the session without saving changes. The editor issues a prompt message that you must respond to prior to actually exiting EDLIN. The prompt message that results from entering the Q command is

```
*Q
Abort edit (Y/N)?
```

In response to the EDLIN prompt, you can type **Y** to quit the editing session. If you do so, no backup file is created and you lose any work you performed on the file in memory. If you do not wish to quit EDLIN, type **N** or any other character and you can continue the present editing session.

T (TRANSFER LINES) Command

The TRANSFER LINES command merges the contents of a specified file into the file you are editing. The line number in the command denotes the position where the merged text will be inserted, with the text being inserted ahead of the specified line. If a line number is omitted, the current line is used. You can also use the period to specify the current line or the # symbol to specify the line after the last line in storage.

As an example of the TRANSFER LINES command, assume you wish to merge the contents of the file STAT.BAS on drive B before line 100 in the file being edited. You would enter the command

```
*100TB:STAT.BAS
```

If you wanted to insert the contents of the file STAT.BAS at the end of the file being edited, you would enter the command

***#TB:STAT.BAS**

A (APPEND) and W (WRITE LINES) Commands

There are two special EDLIN commands that are meaningful only if the file being edited is too large to fit into memory. These are the APPEND LINES and WRITE LINES commands.

Invoking the APPEND LINES command causes *n* lines from the disk to be appended to the file in memory that is presently being edited. Because as much of the file as possible was previously read into memory for editing, you must use the WRITE LINES command to transfer a portion of the file in memory to diskette prior to appending lines to the file in memory.

The WRITE LINES command causes *n* lines in memory, beginning with line number 1, to be written to the diskette. Similar to the APPEND LINES command, this command is only meaningful if the file being edited is too large to fit in memory.

The APPEND LINES command operates until available memory reaches 75 percent of capacity, whereas the WRITE LINES command operates until available memory is less than 25 percent full, that is, if the number of lines is not specified in an APPEND LINES command, lines are appended until available memory is filled to 75 percent of capacity. Similarly, if the number of lines in a WRITE LINES command is not specified, lines are written until available memory is less than 25 percent of capacity. If available memory is already less than 25 percent of capacity, issuing a WRITE LINES command does not cause any action.

8 Batch File Operations

Batch files both automate operations and significantly boost the productivity of DOS and its users. With batch files you can customize DOS for other computer users, simplify complex systems, and perform other operations to facilitate the use of a personal computer.

A batch file consists of a series of DOS commands that are executed when a single command referencing the name of the file is entered. Because you normally test the operation of a batch file prior to using it on a permanent basis or providing it to others, its use eliminates many common causes of DOS operational problems. Such problems can include users typing complex DOS commands incorrectly, performing unintentional DOS operations, or lacking the ability or training necessary to perform DOS related operations.

This chapter first examines ways to create and use batch files. Then, using this information as a base, it examines the operation of specific batch commands and shows how to create several batch files to boost the productivity of your personal computer. In concluding, this chapter explores a special type of batch file designed to hold commands used to configure your computer system. This examination includes the operation and utilization of configuration commands that can be placed in this special batch file.

Creating and Using Batch Files

A batch file consists of ASCII text that represents one or more DOS commands. Each command occupies a separate line in the file. To identify the file as a batch file, its extension must be .BAT. However, to execute the batch file you can enter the name of the file without its extension on the DOS command line. When you do so, DOS searches the current directory for a .COM, .EXE, or .BAT file with a name that matches the name you entered. You should therefore avoid assigning a name to a batch file that duplicates the name of a DOS command.

Because a batch file is restricted to ASCII text, it must be created by a process that generates that type of data. One of the most common methods of creating batch files is to use the DOS COPY command to copy data from the console to your file. As an illustration of the use of the COPY command, consider the following example, which creates the batch file named WP.BAT.

```
COPY CON: WP.BAT
CD \WP
```

WRITE
**CD **
^Z

In this example you must enter a Ctrl+Z multikey combination, pressing the Ctrl and Z keys to terminate the COPY operation. Once you do so, the data entered from the console is written onto the file WP.BAT. Then, by simply entering the command WP, three separate command line entries will be executed.

The first line entry in the WP.BAT file changes the directory to WP. The next line entry, WRITE, is assumed to be a command entry that results in the execution of a word processing program with that name. Finally, the third line entry changes the current directory to the root directory. Although this example is simple, it illustrates how you can create a program that, while transparent to the user, navigates through a hierarchical directory structure to execute a program and then returns to the root directory once the use of that program is terminated. Thus, batch files can be extremely useful in simplifying operations for users not well versed in the use of the operating system.

Although the DOS Shell enables you to easily navigate through a directory structure, you must use batch files to predefine a sequence of operations. Thus, under the DOS Shell you can create one or more batch files to further simplify computer operations. Then a PS/2 user can simply move the section cursor over a batch file's name and press the **Space bar** to select it, after which invoking the `Open (start)` action from the File pull-down will initiate the predefined sequence of operations.

Instead of using the COPY command to create batch files, you can create them with the EDLIN text editor (described in Chapter 7) that is included with DOS, or you can use most word processors. If you use a word processor, make sure that the resulting data is written onto disk as an ASCII file and not as the word processor's normal binary file or as an encoded text file. Otherwise, DOS will not be able to correctly interpret the information contained in the file.

Replaceable Parameters

One of the most helpful features incorporated in a batch file is the ability to use replaceable parameters. DOS enables up to 10 replaceable parameters to be used at a time within a batch file. These parameters are replaced by data supplied by the computer operator in the DOS command line when the batch file is executed.

The replaceable parameters that can be specified must be labeled %0 through %9, with %0 always replaced by a drive specifier, if required, and the filename of the batch file. As an example of replaceable parameters, consider the following batch file.

```
A>COPY CON: WRITE.BAT
CD \WRITE
COPY %1.DAT %1.BAK
WRITE %1
```

Suppose the previously created batch file was created by entering the DOS command line

WRITE JANEMEMO

Here JANEMEMO is substituted for the replaceable parameter %1. Thus, after the directory is changed to WRITE, the batch program substitutes JANEMEMO for %1, resulting in the file JANEMEMO.DAT being copied to the "backup" file JANEMEMO.BAK. Then, the batch program automatically executes a word processing program named WRITE, using the file JANEMEMO.DAT as input.

If you change the COPY command in the previously illustrated batch file to COPY %1.DAT %2.BAK, two filenames would be required to be entered in the DOS command line. In this instance, entering WRITE JANEMEMO JANEBKUP would result in the sequential substitution of JANEMEMO for %1 and JANEBKUP for %2.

Batch Commands

When DOS 1.0 was introduced, it included only two commands designed specifically for use in batch files—REM and PAUSE. DOS 2.0 added five additional batch commands—ECHO, FOR, GOTO, IF, and SHIFT, as well as the undocumented ability to include comments by prefixing line entries with a period (.). This undocumented capability is also known as the *dot command.*

When DOS version 3.0 was released, the batch commands remained essentially unchanged. However, the undocumented dot command was eliminated. The only major change to batch commands occurred with the introduction of DOS version 3.3, which added the CALL command. This batch command enables you to program one batch file and to execute another batch file (that is, it acts like a subroutine call in conventional programming). Then, when execution of the CALLed batch file is complete, control is passed back to the original batch file (just as a subroutine returns control to a main program).

Table 8.1 indicates the batch commands that are supported by different versions of DOS. You can use this table to determine whether you should obtain a more recent version of the operating system for older IBM PCs, PC XTs, and PC ATs if you develop batch files on a PS/2 that will also be used on those computer systems.

Table 8.1
DOS Batch Command Support

Batch Command	DOS Support			
	1.X	2.X	3.0	3.3/4.0
REM	yes	yes	yes	yes
ECHO	no	yes	yes	yes
PAUSE	yes	yes	yes	yes
FOR	no	yes	yes	yes
GOTO	no	yes	yes	yes
IF	no	yes	yes	yes
SHIFT	no	yes	yes	yes
CALL	no	no	no	yes

REM Command

The REM command is an abbreviation for REMark. It documents your batch files by including comments at appropriate locations. The format of this command, which is only used in batch files, is

```
REM [remark]
```

As an example of the use of this command, try modifying your previously created WP.BAT file. The following example illustrates how you might identify each line in the batch file that performs an operation.

```
REM Change to WP directory
CD \WP
REM Execute word processing program
WRITE
REM Return to root directory
CD \
```

For clarity, many persons prefer to group REM commands together. Thus, as an alternative the file might be created using REM commands as follows:

```
REM Change to WP directory
REM Execute word processing program
REM Return to root directory
CD \WP
WRITE
CD \
```

The DOS commands used in the sample three-line batch file are essentially self-explanatory and for many persons probably do not require documentation. However, you'll be saved hours of future effort if you have to modify a previously created batch file and the documentation is explicit in explaining the operation of complex statements in the file.

ECHO Command

This command can be used to enable or disable the display of DOS commands as they are executed from within a batch file. The format of this command is

```
ECHO [{ON }] [message]
     [{OFF}]
```

When ECHO is entered with no parameters, the command causes the current ECHO state to be displayed as indicated here.

```
A>ECHO
Echo is ON
```

ECHO ON is the default state when your system is powered on or a system reset is performed. By issuing an ECHO OFF command, you instruct DOS to inhibit the

display of the DOS prompt and commands on the screen as they are executed from within a batch file. Similarly, an ECHO ON command displays the DOS prompt and commands on the screen as they are executed from within a batch file. If you include a message in the ECHO command, it is displayed regardless of the ECHO state. As an alternative to ECHO OFF, you can prefix any command with an @ symbol to inhibit its display. Unfortunately, this does not prevent DOS from displaying messages generated by the execution of commands, such as `x File(s) copied`. To suppress these messages, you should direct the output to the NUL device by adding **>NUL** after any DOS command that normally generates a message on the display.

PAUSE Command

The PAUSE command can be used to temporarily suspend the execution of a batch file. The format of this command is

```
PAUSE [remark]
```

When the PAUSE command is executed, it first displays any remark included in the command line. Then, the command temporarily suspends the execution of the batch file it is included in and displays the message `Strike a key when ready...` on a new line. Once a key is pressed, the batch file resumes execution at the next line following the line containing the PAUSE command. The following short batch program illustrates how the PAUSE command can be used to provide information to the operator of the personal computer. Note that FORMAT is a dangerous command to play with unless you absolutely know what you are doing.

```
COPY CON: FORMAT.BAT
ECHO OFF
XFORMAT A:
```

Assuming you previously renamed the DOS FORMAT file as XFORMAT and named your batch file FORMAT, the result of entering your new FORMAT command is

```
C>FORMAT
Insert new diskette for drive A:
and strike ENTER when ready
```

Note that the preceding batch file changed the use of the DOS FORMAT command to restrict its usage to diskettes in drive A. Later this chapter examines in detail several methods you can use to prevent formatting the hard disk inadvertently.

FOR Command

The FOR command enables the iterative execution of a specified DOS command. The format of this command is

```
FOR %%variable IN(file set)DO command
```

When the FOR command is executed, a single-letter *variable* included in the *command* is set sequentially to each member of the *file set*. After the *variable* is set to a member of the *file set*, the command following the DO specification is executed.

This process repeats until all file set members are processed. As an example of the use of FOR, assume you want to obtain a listing of all the batch files and data files under the subdirectory DBASE. To accomplish this, you could create a batch file like this:

```
COPY CON: LSTDBASE.BAT
CD \DBASE
FOR %%X IN(*.BAT *.DAT)DO TYPE %%X>LPT1:
^ZCD \
```

Once the batch file named LSTDBASE is created, entering its filename causes the directory to be changed to DBASE. Then, the FOR command assigns the variable X to the file *.BAT, which is actually a potential set of files, because the asterisk global character is included in the file specification. Next, the TYPE command is executed for all files whose extension is .BAT, after which the variable X is assigned to the file set *.DAT and the process of printing the contents of all files whose extension is .DAT occurs. As indicated in this example, global filename characters can be included in the FOR command to increase the power and utility of this batch command.

GOTO Command

The GOTO command performs branching operations. This command, as you will soon see, can be used with any combination of commands or conditional operators.

The format of the GOTO command is

```
GOTO [:]label
```

The *label* is required here because it indicates a destination point in the batch file to which branching should occur.

The *label* is a string of up to eight characters, which must be prefixed by a colon (:) when the label is used as a branch entry point. When used in this manner, the label is contained on a separate line in the batch file. The following example illustrates the use of the GOTO command, as well as the inclusion of a label in a batch file.

```
:START
PAUSE Insert the disk to be copied in drive A
COPY A:*.* C:\LOTUS\JOHN
GOTO START
```

In the preceding example a continuous loop through the batch file occurs, so the computer operator must press Ctrl+Break to terminate the batch job and return to the DOS prompt level of operation. When you examine the IF command you will use that command to construct a conditional branching example. This example more fully demonstrates the capability of the GOTO command that occurs when its use is combined with another batch command.

IF Command

The IF command is one of the most useful batch commands, because it enables you to incorporate conditional branching into a batch file. The format of this command is

```
IF [NOT] condition command
```

where the *condition* parameters are

```
ERRORLEVEL number
String1 == String2
EXIST filespec
```

When the IF command is executed, the specified *condition* in the *command* is evaluated. If the condition is True, the command specified after the condition is executed. If the condition is False, the command specified after the condition is ignored and control is passed to the next line in the batch file.

The ERRORLEVEL number refers to the value of an optional exit code that a previously executed program may have set. When a program completes its execution correctly, it returns to DOS with an unset error flag. If the program terminates incorrectly, its error flag is set, causing DOS to examine a specific location to determine what the error code means.

Four DOS commands that return error codes if they terminate with an error that can be checked are FORMAT, REPLACE, BACKUP, and RESTORE. As an example of the use of error codes, assume you are creating a batch program file to automate BACKUP operations. In using an ERRORLEVEL X conditional operator in the IF command, the command will be executed if the returned error code is X or higher. Thus, the following simple batch file demonstrates the use of the ERRORLEVEL operator in the batch IF command.

```
BACKUP
IF ERRORLEVEL 1 ECHO!Backup Aborted!
```

The condition `String1 == String2` in an IF command is true when both strings are identical. Unlike normal string comparisons, string values used in an IF command cannot include data redirection characters or such command line delimiters as a comma, equal sign, colon, or semicolon. As an example of the use of a string comparison condition in IF commands, consider the batch file in Listing 8.1, which includes the COPY command in line 1 that creates the batch file.

Listing 8.1 LISTDISK.BAT

```
COPY CON: LISTDISK.BAT
IF %1 == LEFT GOTO DRIVEA
IF %1 == RIGHT GOTO DRIVEB
ECHO YOU OBVIOUSLY DON'T KNOW WHAT TO DO
GOTO END
:DRIVEA
DIR A:>LPT1:
GOTO END
:DRIVEB
DIR B:>LPT1:
:END
```

This batch file enables you to simply enter the name of the file followed by the word LEFT or RIGHT to obtain a directory listing of an appropriately located drive. When this batch file is executed, the first entry in the command line following LISTDISK is assigned to the replaceable parameter %1. Thus, if the command line entry is

A>LISTDRIVE LEFT

the replaceable parameter %1 would be assigned the value LEFT. Because the first IF command is true, a branch to the label DRIVEA occurs in the program. The line after that label contains the DOS command DIR A:, causing the directory of that drive to be printed, after which a branch to the end of the batch file occurs. Similarly, if you enter the command

LISTDISK RIGHT

the directory of the diskette in drive B is printed. Although this is a trivial example, it illustrates the capability of using string comparisons in an IF command.

The last condition that you can test in an IF command is the existence or nonexistence of a file specification. By using the conditions EXIST or NONEXIST, you can test for the existence of a file in the default directory of a specified drive. The following portion of a batch file illustrates how you can test for the presence of a required file on a diskette in a specific drive and display an appropriate message if it is not found.

```
:START
PAUSE Insert disk labeled XYZ in drive A
IF EXISTS A:JAN88.WKS GOTO DOIT
PAUSE Wrong disk inserted into drive A
GOTO START
:DOIT
REM Program continues
```

SHIFT Command

The SHIFT command enables you to use more than 10 replaceable parameters in a batch file. When included in a batch file, this command changes the order in which the parameters %0 through %9 are replaced. That is, each SHIFT command causes every parameter to be shifted to the left. Because this may sound confusing, clarify the use of this command by creating a batch file that uses the SHIFT command. The contents of the file, named SHOSHIFT.BAT, appear in Listing 8.2.

Listing 8.2 SHOSHIFT.BAT

```
ECHO OFF
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

Next, assume you entered the following command line:

SHOSHIFT A B C D E F G H I J K

which displays:

```
A>ECHO OFF
A B C D E F G H I
B C D E F G H I J
C D E F G H I J K
```

As indicated in this example, you can use the SHIFT command to extend the number of replaceable parameters that can be used in a batch file. Although 10 replaceable parameters are usually sufficient for most applications, for creating complex batch processes the SHIFT command can be extremely useful.

CALL Command

Added with the introduction of DOS 3.3, this batch command enables a second batch file to execute without requiring the first one to terminate. The format of this command is

```
CALL [d:][path]filename
```

The following portion of a batch file named RUN.BAT illustrates the use of the CALL command.

```
IF %1 == LOTUS CALL C:LOTUS
IF %1 == DBASE CALL D:DBASE
REM Program continues
```

By entering the DOS command line **RUN LOTUS**, the file LOTUS on drive C is called. Similarly, entering the DOS command line **CALL DBASE** invokes the batch file named DBASE located on drive D.

The AUTOEXEC.BAT File

The AUTOEXEC.BAT file is a special batch file you can use to automatically execute DOS commands. This special file is searched for by DOS each time your computer is powered-on or a system reset is performed. If this file is located in the root directory of the drive from which DOS is initialized, it is automatically executed.

The primary use of an AUTOEXEC.BAT file is to execute a series of DOS commands that are always initially required for the operation of your computer system. One example of the usefulness of AUTOEXEC.BAT is to display a menu system to facilitate the use of the computer by personnel who are not educated in the DOS commands. Later this chapter presents several menu systems that can be generated through the use of an AUTOEXEC.BAT file. Even if you are using DOS 4.0, you may wish to develop a customized menu system to facilitate the operations of your organization.

Boosting Productivity

Batch files can perform a variety of tasks related to files and subdirectories and can make the interface between the user and the computer much simpler. As an example, instead of issuing separate commands to first change the DOS current directory and then invoke a program such as Lotus 1-2-3, you might simply create a batch file to run 1-2-3. Then, issuing one batch filename would automatically change the directory and execute the desired program.

This section describes the use of batch programs for operations that are normally performed on files and subdirectories. Some of the major topics covered include the development of several format protection schemes, a method to enhance disk copying operations on systems that have only one disk drive, and methods to create your own help files, as well as several menu systems to simplify fixed disk usage.

Because batch files have names that are assigned by the user, it is important that your batch files do not duplicate names of DOS commands. It is also crucial not to create two batch files with the same name to perform different functions.

Format Protection Schemes

Because the formatting of a disk destroys any previously recorded data, a mechanism for preventing inadvertent formatting is highly desirable. This section explains two format protection schemes that can be used to satisfy different operational requirements.

If you have a fixed disk system, the FORMAT.EXE file from the DOS distribution diskette was probably copied to the root directory or to a subdirectory named UTILITY or DOS. If another person using the computer is at the C> prompt level and enters the command FORMAT, the resulting action depends on the version of DOS used. Although recent versions of DOS from 3.0 onward provide a warning, earlier versions of DOS simply initiate the format operation, destroying any data previously recorded on the hard disk. Even when a warning message is displayed, if it is late at night or if you or another user are distracted while performing this operation, it is quite possible that entering the wrong response to a warning message can destroy months or years of effort.

To prevent the inadvertent formatting of the fixed disk you can first rename the FORMAT.EXE file to a less obtrusive name. Because the format process prepares a disk for use, you could rename it as indicated here:

```
RENAME FORMAT.EXE PREPARE.EXE
```

Next, you could create a one-line batch file to restrict formatting to drive A. To create this batch file, use the name FORMAT for the file as indicated here:

```
COPY CON: FORMAT.BAT
PREPARE A:%1 %2 %3 %4 %5 %6
```

In this example, entering the command FORMAT invokes the execution of a batch file with that name. This file causes the FORMAT.EXE program that was previously renamed PREPARE.EXE to be executed. Up to six replaceable parameters are passed to the renamed formatting program. This means you can pass six of the eight parameters

that can be specified in the FORMAT command. In addition, the drive specifier A: is always passed to the formatting program, restricting the use of the program to formatting diskettes in drive A.

The following illustrates the execution of the previously created batch file and the resulting execution of the renamed FORMAT command:

```
C>format

C>PREPARE A:

Insert new diskette for drive A:
and strike ENTER when ready
```

Because the previously created one-line batch file executes the command PREPARE, it is displayed on the user's screen. If you feel this may cause confusion, you could prefix the PREPARE command with an ECHO OFF command. The modification of the previously created batch file and its execution is

```
C>copy con: format.bat

echo off
prepare a:%1 %2 %3 %4 %5 %6
^Z
     1 File(s) copied

C>format

C>echo off
Insert new diskette for drive A:
and strike ENTER when ready
```

Considering Device Names

Because many users are familiar with device names, they would probably enter the command FORMAT A: to format a diskette in drive A. If this entry occurs as you use the previously created batch file, the error message `Invalid parameter` is displayed and the batch program terminates. This happens because the drive specifier A: was previously included in the batch file. To permit additional flexibility in using the FORMAT command, you can rename FORMAT.EXE as PREPARE.EXE and create the batch file listed in Listing 8.3 to execute the renamed FORMAT program.

Listing 8.3 Considering Device Names

```
ECHO OFF
IF %1==A: GOTO OK
IF %1==a: GOTO OK
IF %1==B: GOTO OK
IF %1==b: GOTO OK
```

```
ECHO This program only formats diskettes in drive A or B
GOTO END
:OK
PREPARE %1 %2 %3 %4 %5 %6
:END
```

This second example is more flexible for many users, because it enables formatting to occur on diskettes in drive A or drive B. In this example, lines two through five test the replaceable parameter for the value A, a, B, or b and result in a branch to the label OK if a match occurs. If no match occurs, the message `This program only formats diskettes in drive A or B` is displayed and a branch to the label END occurs.

Enhancing Single-Drive Copying Operations

Although the PS/2 Model 25 and similar computers containing one floppy diskette drive are fine machines, they are not easily used for diskette copying operations. When you need to use the DISKCOPY A: B: command or its near equivalent, COPY A:*.* B:, DOS uses your one physical drive as two logical devices. Due to this, you will be treated to what is known as the "floppy shuffle" as you are alternately prompted to insert source and target diskettes as each file is copied. If you have to duplicate a large number of files on the diskette, you can conceivably spend 20 minutes or more performing the floppy shuffle.

To alleviate the floppy shuffle, you will see how to create a batch file that automatically uses a portion of the capacity of your fixed disk as a second diskette drive. This batch file creates a temporary directory named TEMPCOPY under the root directory and copies all files from the diskette to be copied to that directory. Next, your batch file prompts the user to place a formatted disk in drive A, copy the contents of the TEMPCOPY directory to the diskette in drive A, erase the contents of the temporary directory, and then remove the directory. Listing 8.4 shows the creation of this batch file using the DOS EDLIN text editor program.

Listing 8.4 FASTCOPY.BAT

```
 1: ECHO OFF
 2: ECHO Place Diskette to be copied in Drive A
 3: PAUSE
 4: MD \TEMPCOPY
 5: COPY A:*.* C:\TEMPCOPY
 6: ECHO Place target Diskette in Drive A
 7: PAUSE
 8: ECHO Enter Y in response to next prompt
 9: ERASE C:\TEMPCOPY\*.*
10: RD C:\TEMPCOPY
```

In the batch file listed, line 1 turns the echoing of executed commands off, and line 2 prompts the user to place the diskette to be copied in drive A. The PAUSE command

in line 3 displays the prompt `Strike a key when ready...`. Once a key is pressed, line 4 creates the directory named TEMPCOPY under the root directory.

Copying the contents of the diskette in drive A to the subdirectory TEMPCOPY results from the execution of the COPY command in line 5. Next, the ECHO command in line 6 prompts the computer user to place the diskette in drive A onto which the files will be copied. The PAUSE statement in line 7 again displays the prompt `Strike a key when ready...` on the screen.

After the files are copied to the diskette, it is assumed that they are of no use on the fixed disk and should be removed. Because the use of the ERASE command with the global asterisk characters for filename and extension generates the DOS message `Are you sure (Y/N)?`, line 8 was included in the batch file to prompt the user how to respond to that message. Then, the execution of line 9 erases all files in the directory TEMPCOPY on drive C once the user enters the letter Y in response to the previously described DOS message. Finally, line 10 removes the directory TEMPCOPY from the fixed disk.

The following interactive example illustrates how to execute the FASTCOPY.BAT batch file. In this example, a diskette containing six files was to be copied. Using FASTCOPY instead of the DOS COPY command reduces the number of diskette insertions on a one drive system from 12 to 2, which illustrates the major advantage in using this batch file.

```
C>FASTCOPY

C>ECHO OFF
Place DISKETTE to be copied in Drive A...
Strike a key when ready...
A:AVG.PIC
A:PERF.WKS
A:WK1.PIC
A:WK2.PIC
A:WK3.PIC
A:WK4.PIC
     6 File(s) copied
Place Target Diskette in Drive A...
Strike a key when ready...
C:\TEMPCOPY\AVG.PIC
C:\TEMPCOPY\PERF.WKS
C:\TEMPCOPY\WK1.PIC
C:\TEMPCOPY\WK2.PIC
C:\TEMPCOPY\WK3.PIC
C:\TEMPCOPY\WK4.PIC
     6 File(s) copied
Enter Y in response to next prompt
Are you sure (Y/N)?Y

C>
```

Creating DOS Command HELP Files

If you or your users operate an IBM PS/2 computer and are not well versed in the use of DOS commands, on-line assistance can be a most helpful feature to provide. In this section you create a batch file named HELP.BAT that enables users to enter the command HELP followed by a DOS command to obtain on-line assistance concerning the use of the specific command.

Although the DOS 4.0 Shell contains a context-sensitive help facility invoked by pressing the F1 key, this facility has many limitations. As an example, for most DOS commands the help facility will refer the computer user to the DOS manual for information concerning the parameters available for use with a command. By creating your own help files, you may be able to substantially improve the use of your PS/2, as you will see by developing a comprehensive ERASE help file that provides the PS/2 user with substantially more information concerning the use of the ERASE command than can be obtained using the DOS Shell help facility.

Listing 8.5 shows the creation of the HELP.BAT file.

Listing 8.5 HELP.BAT

```
COPY CON: HELP.BAT
ECHO OFF
IF EXIST %1.HLP GOTO LOCATE
IF %1==0 GOTO HELP
ECHO HELP NOT AVAILABLE FOR %1
GOTO END
:LOCATE
TYPE %1.HLP
GOTO END
:HELP
TYPE HELP.HLP
:END
```

After turning ECHO off, the IF command checks for the existence of a file whose name is the first replaceable parameter used when you run the program and whose extension is .HLP. If the file exists, a branch to the label LOCATE occurs, after which the TYPE command displays the file whose name was entered in the command line when the batch file was initiated and whose extension is .HLP.

If the name of the DOS command assigned to the replaceable parameter %1 does not exist with the extension .HLP in the current directory, the second IF command will be executed. This IF command is included in the batch file to avoid the need to type HELP HELP in order to display a master HELP file that explains how to obtain help for selected DOS commands. Instead, entering HELP by itself results in the replaceable parameter %1 being assigned a null value of "". Because this value results in a match in the string comparison, a branch to the label HELP occurs. Then, the next line in the batch file displays the HELP.HLP file on the screen. If you plan to have more than one screenful of help on a topic, incorporate the MORE command so that the text doesn't scroll off the screen before it's read.

If you create separate ASCII files that explain the use of such DOS commands as COPY, DIR, FORMAT, and ERASE, you only have to enter the command

HELP *command*

where *command* is a DOS command for which there is an ASCII file explaining the use of the command. Then, the contents of that file are displayed to give on-line assistance.

Listing 8.6 shows how to create the file ERASE.HLP, using the EDLIN text editor contained on the DOS diskette. The file contains text that explains the use of the ERASE command. The following display shows exactly what you'll see on the screen if you enter HELP ERASE to obtain information on the use of the DOS ERASE command.

Listing 8.6 ERASE.HLP

```
1:*The ERASE command is used to erase a file from disk.
2:*
3:*The format of ERASE is:
4:*                                    ERASE [d:][path]filename[.ext]
5:*     where:
6:*               [d:]       - disk drive where file is located
7:*               [path]     - directory path to the file
8:*               filename   - name of the file to erase
9:*               [.ext]     - filename extension (if present)
10:*
11:*  Examples:
12:*               ERASE B:\123\GRAPH1.PIC
13:*
14:*               ERASE SAMPLE.HLP
```

```
A>help erase

A>echo off
The ERASE command is used to erase a file from disk.

The format of ERASE is:
                   ERASE [d:][path]filename[.ext]
where:
                      [d:]       - disk drive where file is located
                      [path]     - directory path to the file
                      filename   - name of the file to erase
                      [.ext]     - filename extension (if present)

    Examples:
                      ERASE B:\123\GRAPH1.PIC

                      ERASE SAMPLE.HLP
```

Expansion to Application Programs

You can also create other ASCII text files with the extension HLP to provide on-line assistance relating to application programs. As an example, assume you have two application programs named BEST and WORST for which you would like to provide users with on-line assistance. You simply create text files with EDLIN named BEST.HLP and WORST.HLP, in the same way in which you created the ERASE.HLP file in Listing 8.6. Then, entering the command HELP BEST (or HELP WORST) causes the HELP.BAT file displaying the contents of BEST.HLP (or WORST.HLP) on the screen.

Master Menus for Fixed Disk Systems

The use of fixed disk (also called hard disk) systems has become much more prevalent recently because of the decreasing cost and increased availability of fixed disk storage devices. Many users today place multiple applications on their fixed disks, resulting in a requirement for an easy way to navigate between applications. You can design a simple batch menu system to assist with this task, or you can create a more complex system that accomplishes a similar function in a more elegant manner. This section examines several methods of creating master menus, beginning with a simple method that helps you navigate among directories.

Suppose you have a PS/2 with a large fixed disk and wish to use 1-2-3, dBASE III, and MultiMate. A simple menu scheme enables users to choose among each of these applications by simply selecting a letter or number from the menu. An example of this menu is

```
                                    GILBERT'S MAIN MENU
=============================================================================

                                        A - LOTUS 123

                                        B - DBASE III

                                        C - MULTIMATE



                                        X - Exit MENU System to DOS


Enter your selection -
```

Once this menu is displayed, the user can simply type the letter of his or her choice and press the **Enter** key to use a selected application. To practice this technique, create files named A.BAT, B.BAT, and C.BAT to begin each of the applications and a batch file called X.BAT to provide a means to exit the MENU system and return to the DOS

prompt. A file called MENU.BAT will be used to display the menu on the screen and give the user a choice. The MENU.BAT file is shown in Listing 8.7.

Listing 8.7 MENU.BAT

```
1:*echo off
2: cls
3: type menu.txt
4: prompt Enter your selection -
```

Line 1 of MENU.BAT turns off the echo of commands to the screen so the user does not see what commands are issued. Line 2 clears the screen, and line 3 places the menu on the screen by typing a file called MENU.TXT. This file is

```
 1:*                                    GILBERT'S MAIN MENU
 2: ============================================================
 3:
 4:                                        A - LOTUS 123
 5:
 6:                                        B - DBASE III
 7:
 8:                                        C - MULTIMATE
 9:
10:
11:
12:                                        X - Exit MENU System to DOS
```

Line 4 of the MENU.BAT file changes the DOS prompt from the standard of default disk drive and greater than symbol to the words `Enter your selection -`. When the menu is placed on the screen, the user is asked to select one of the choices on the menu, as shown:

```
                                       GILBERT'S MAIN MENU
============================================================

                                          A - LOTUS 123

                                          B - DBASE III

                                          C - MULTIMATE



                                          X - Exit MENU System to DOS


Enter your selection -
```

The X.BAT file exits from the menu and returns control to the DOS system. In reality, however, control is still at the DOS level but the prompt has been changed. All the X.BAT file in Listing 8.8 has to do is return the prompt to its standard symbol, the default disk drive and the greater than symbol (>).

Listing 8.8 X.BAT

```
1:*echo off
2: cls
3: prompt $n$g
```

Line 1 simply turns the echo of commands to the screen off. Line 2 clears the screen, and line 3 changes the DOS prompt back to its standard symbol, the default disk drive followed by the greater than symbol.

The A.BAT file invokes 1-2-3. This batch file is illustrated in Listing 8.9.

Listing 8.9 A.BAT

```
1:*echo off
2: cls
3: cd \123
4: lotus
5: cd \
6: menu
```

Line 1 turns the echo of commands to the screen off, and line 2 clears the screen. Then, line 3 uses the DOS Change Directory (CD) command to change the current directory to the subdirectory containing the 1-2-3 software. In this example, it is assumed that 1-2-3 resides in the directory named 123. Next, line 4 executes the 1-2-3 software. When the user exits 1-2-3, DOS executes the next command in the batch file, called LOTUS, resulting in line 5 changing the current directory back to the highest level or root directory. Finally, line 6 issues the MENU command, which executes the MENU.BAT file and places the master menu back onto the screen.

The B.BAT file invokes the dBASE III program. This batch file appears in Listing 8.10. Notice that it is structured exactly the same as the A.BAT file, except that line 3 changes the current directory to the DBASE subdirectory while line 4 executes the dBASE software. Lines 5 and 6 are used to return to the master menu.

Listing 8.10 B.BAT

```
1:*echo off
2: cls
3: cd \dbase
4: dbase
5: cd \
6: menu
```

The last file required to create the simple menu is the C.BAT file. This file is used to invoke MultiMate word processing software and is illustrated in Listing 8.11. Note

that it is the same as the A.BAT and B.BAT files, except that line 3 changes the current directory to the MULTIMATE subdirectory while line 4 executes the MultiMate software.

Listing 8.11 C.BAT

```
1:*echo off
2: cls
3: cd \mm
4: mm
5: cd \
6: menu
```

Using BASIC

A different method of developing a menu system uses the BASIC interpreter that comes with DOS. You start the interpreter by issuing the command BASIC, or BASICA for the advanced BASIC, or GWBASIC for the BASIC provided with many versions of MS-DOS developed for clones.

Listing 8.12 shows an example of a BASIC program for controlling software selections on Jane's PC. This program has three options from which to select: word processing, spreadsheet, or database applications. The BASIC SHELL command is used to execute a batch program corresponding to the selection made from the menu. When you terminate the word processor, spreadsheet, or database function, control is returned to the BASIC program and the menu is again displayed on the screen.

Listing 8.12 BASIC Language Program for Controlling Software Selection

```
10 REM ************************************************************
20 REM *
30 REM * Main Menu for JANE's PC
40 REM *
50 REM ************************************************************
60 CLS
70 PRINT "                          JANE's PC MAIN MENU"
80 PRINT "------------------------------------------------------------------"
90 PRINT
100 PRINT
110 PRINT "             1 - WORDPROCESSING"
120 PRINT
130 PRINT "             2 - SPREADSHEET"
140 PRINT
150 PRINT "             3 - DATABASE"
160 PRINT
170 PRINT
180 PRINT
190 PRINT
200 INPUT "Enter the NUMBER of your choice... ";N
```

```
210 IF N=1 THEN SHELL "C:\BATUTIL\WP.BAT"
220 IF N=2 THEN SHELL "C:\BATUTIL\SS.BAT"
230 IF N=3 THEN SHELL "C:\BATUTIL\DB.BAT"
240 GOTO 60
250 END
```

Lines 10 through 50 are simply remarks within the program using the BASIC REM statement. These lines provide anyone who looks at the program with information the author of the program feels is important. In this example, it's simply a remark that this is the main menu for Jane's PC. Other information could be added, such as the name of the program and the date it was created, as well as the name of the author or programmer.

Line 60 is the BASIC CLS statement that simply clears the screen on the PC similarly to the DOS CLS command. Lines 70 through 190 use the PRINT command to place the menu on the PC screen. Line 200 uses the INPUT command to obtain a response from the user. The INPUT command first asks the user to enter a number of his or her choice from the menu and waits for the user to enter a number into the variable N. Line 210 checks whether the number entered is a 1 and if so executes the SHELL command to run a batch file named WP.BAT under the BATUTIL subdirectory on the C drive. Lines 220 and 230 work in a manner similar to line 210. That is, they check if the number entered is a 2 or 3 and execute batch files named SS.BAT and DB.BAT, respectively, on encountering the predefined number. If the user enters anything other than a 1, 2, or 3 the GOTO command at line 240 is executed. This GOTO causes the program to start over at line 60, which first clears the screen and then places the menu back onto the screen. Line 250 is an END command that indicates the end of the program.

Although the DOS ECHO command performs the same general function as the BASIC print statement, BASIC provides greater flexibility for performing the required tasks. As an example, BASIC programming statements enable you to compare keyboard input against predefined values and to then perform branching operations based on the value of the input. In comparison, batch file processing only permits data input in the form of replaceable parameters that are entered when the batch file is executed. Thus, you can easily construct a BASIC program to accept keyboard input to perform different operations, whereas attempting to do the same with batch files can be either extremely difficult or impossible.

A second advantage in the use of BASIC for creating menus is the ease of screen control in that language. With the BASIC COLOR statement you can easily set foreground, background, and border colors, as well as perform highlight and blinking operations. Whereas these operations can also be performed from within batch files, many readers familiar with BASIC will prefer to perform those operations from within that language. Thus, the use of the SHELL command is the key to obtaining the ability of increased menu programming flexibility from within BASIC.

The SHELL Command

The SHELL command runs a command or batch file from within a BASIC program and is the key to how the menu works properly. In essence, the BASIC program is

simply put to "sleep" while the batch file or DOS command is executed. When the batch file is finished with its work or the DOS command is completed, the DOS command EXIT is used to return to the BASIC program, "wake" it up, and allow it to continue operating at the point where it was last running.

Figure 8.1 shows what the BASIC program looks like on the screen when it is executed.

Suppose the program is saved on your hard disk under the name MENU.BAS. To run the program, simply enter the command

```
BASIC MENU
```

This causes the BASIC interpreter to start and automatically loads and executes the program MENU.BAS. Of course, the BASIC interpreter must be in your current directory or pointed to by the DOS PATH command, whereas the program MENU.BAS must also be in your current directory or in a directory pointed to by the DOS PATH command.

Instead of entering the command yourself, you can set up a system to automatically start this menu by creating an AUTOEXEC.BAT file as shown:

```
1: echo off
2: path = \DOS
3: BASIC MENU
```

This file turns off the echo of commands to the screen in line 1. In line 2, the DOS PATH command sets up search paths to the subdirectory \DOS, which is where the BASIC interpreter is assumed to reside. Then, line 3 executes the BASIC interpreter and the MENU.BAS program.

Listing 8.13 illustrates the contents of the WP.BAT file.

Listing 8.13 WP.BAT File

```
1:*echo off
2: cd \mm
3: mm
4: exit
```

Figure 8.1
Resulting BASIC Program Display

```
                              JANE's PC MAIN MENU
------------------------------------------------------------------------

                    1 - WORDPROCESSING

                    2 - SPREADSHEET

                    3 - DATABASE


Enter the NUMBER of your choice... ?
```

Line 1 turns off echoes of commands to the screen. Line 2 uses the DOS change directory command to set the \MM subdirectory as the current directory, and line 3 executes the word processor program. Line 4 is the DOS EXIT command, which is executed when the word processor is finished.

This EXIT command returns control to the BASIC program that called the batch file and is necessary, because the SHELL command was used within the program. Similarly, the SS.BAT and DB.BAT files are shown in Listings 8.14 and 8.15. These files operate exactly like the WP.BAT file but use different subdirectories and call a spreadsheet and database, respectively. That is, the SS file changes the directory to 123, whereas the DB file changes the directory to DBASE. Then, each file issues the appropriate command to initiate the program in each subdirectory.

Listing 8.14 SS.BAT File

```
1:*echo off
2: cd \123
3: lotus
4: exit
```

Listing 8.15 DB.BAT File

```
1:*echo off
2: cd \dbase
3: dbase
4: exit
```

Increasing Menu Functionality

To add additional functionality to the BASIC menu, increase the documentation with REM statements and add an error routine to catch simple errors. Listing 8.15 shows an updated BASIC program that retains the name MENU.BAS.

Listing 8.15 Revised BASIC Program

```
10 REM ************************************************************
20 REM *
30 REM * Program name:   MENU.BAS
40 REM * Author:         Tom Domore
50 REM * Last update:    December 12, 1987
60 REM *
70 REM * This program is the main menu for JANE's PC
80 REM *
90 REM ************************************************************
100 CLS
110 PRINT "                          JANE's PC MAIN MENU"
120 PRINT "------------------------------------------------------------"
130 PRINT
```

```
140 PRINT
150 PRINT "              1 - WORDPROCESSING"
160 PRINT
170 PRINT "              2 - SPREADSHEET"
180 PRINT
190 PRINT "              3 - DATABASE"
200 PRINT
210 PRINT
220 PRINT
230 PRINT "              0 - EXIT...Return to DOS"
240 PRINT
250 PRINT
260 INPUT "Enter the NUMBER of your choice... ";N
270 IF N=0 THEN CLS :SYSTEM
280 IF N=1 THEN SHELL "C:\BATUTIL\WP.BAT"  :GOTO 100
290 IF N=2 THEN SHELL "C:\BATUTIL\SS.BAT"  :GOTO 100
300 IF N=3 THEN SHELL "C:\BATUTIL\DB.BAT"  :GOTO 100
310 PRINT
320 PRINT "*** ERROR ***  -->  You must enter a NUMBER between 0 and 3..."
330 PRINT
340 INPUT "Press <ENTER> or <RETURN> key to try again...";X$
350 GOTO 100
360 END
```

Lines 10 through 90 are all remarks, but this time the program has added information that may be beneficial in a business environment. The program name is MENU.BAS, the author is Tom Domore, and the date the program was last changed was December 12, 1987. This information could be used by someone who later was given the task of changing the menu on all PCs in the company to add additional menu options for such applications as communications or graphics.

Line 100 still clears the screen, while lines 110 through 250 use the PRINT command to put the menu on the screen. Notice that the option `0 - EXIT...Return to DOS` was added to the menu. You may have a user who wishes to work at the DOS level and is knowledgeable about PCs. This option offers an easy and "clean" way to safely exit the BASIC MENU program. Line 260 uses the INPUT command to request the user to select from the menu and places the value selected into the variable `N`. Line 270 is used to check for the selection of 0, indicating that the user wishes to exit to DOS. If the choice is 0, the screen is cleared with the CLS command, and the SYSTEM command returns control to the DOS system. SYSTEM is actually a BASIC command that closes the BASIC program, stops the BASIC interpreter, and turns control back over to the Disk Operating System.

Lines 280 through 300 each checks for a selection from the menu and, based on the number selected, executes a batch file using the SHELL command. When the batch file is completed, the next command after the SHELL command is executed and, in this case, it is the GOTO 100 command. This returns the program to the clear screen routine and places the menu back on the screen so the user can make another selection.

This simple modification to the program now accepts any selection that is not a number 0 through 3 to fall through the program to line 310. Lines 310 through 340 print an error message on the screen telling the user to enter a number between 0 and 3. The INPUT command at line 340 simply allows the program to pause long enough for the user to read the error message before line 350 is executed. Line 350 returns control to line 100, which clears the screen and places the menu back on the screen. An example of the execution of this program is shown in Figure 8.2.

There is still one problem with this program as it is written. What if someone enters a letter or special character and not a number as planned for? In this situation, BASIC is looking for a numeric value (or number) for the variable N in line 260, and any character other than a numeric value triggers a BASIC error that displays a question mark and prompts the user to `Redo from start`, meaning to enter the number again. This message is not very meaningful to someone who is not familiar with computers and the BASIC programming language, raising the potential of a degree of user confusion if it is displayed. Figure 8.3 illustrates the program's response to a user entering **QUIT** instead of 0 for EXIT. Note that the `?Redo from start` message generated by

Figure 8.2
Execution of Revised BASIC Program

```
                              JANE's PC MAIN MENU
------------------------------------------------------------------------

                    1 - WORDPROCESSING

                    2 - SPREADSHEET

                    3 - DATABASE


                    0 - EXIT...Return to DOS

Enter the NUMBER of your choice... ? 7

*** ERROR ***  -->  You must enter a NUMBER between 0 and 3...

Press <ENTER> or <RETURN> key to try again...?
```

Figure 8.3
BASIC Language Generated Error Message

```
                              JANE's PC MAIN MENU
------------------------------------------------------------------------

                    1 - WORDPROCESSING

                    2 - SPREADSHEET

                    3 - DATABASE


                    0 - EXIT...Return to DOS

Enter the NUMBER of your choice... ? QUIT
?Redo from start
Enter the NUMBER of your choice... ?
```

the BASIC interpreter is followed by the program displaying the `Enter the NUMBER of your choice...?` message a second time. To the novice user, is this a request to start all over or should the user simply enter a number?

To eliminate the potential confusion, you should eliminate the possibility of the error message being generated. Change line 260 of the program to accept any character, whether numeric or alphabetic, using a string variable `N$` in the statement. Then, if a string variable is used, lines 270 through 300 of the program must be modified to check for the numbers 0 through 3 in the form of string values. The changes to lines 260 through 300 of the program previously listed in Listing 8.15 are

```
260 INPUT "Enter the NUMBER of your choice...";N$
270 IF N$="0" THEN CLS :SYSTEM
280 IF N$="1" THEN SHELL "C:\BATUTIL\WP.BAT" :GOTO 100
290 IF N$="2" THEN SHELL "C:\BATUTIL\SS.BAT" :GOTO 100
300 IF N$="3" THEN SHELL "C:\BATUTIL\DB.BAT" :GOTO 100
```

As a result of the modification of the program to accept keyboard input as strings, let us again execute the program and enter invalid data. Figure 8.4 illustrates the execution of the BASIC program designed to accept string data. Note that the improper keyboard entry of QUIT does not result in the generation of the BASIC `?Redo from start` error message. Because `N$` does not equal the string 1, 2, or 3, the error message at line 320 in the program is executed. Then, once the user presses the **Enter** key the program branches to line 100, clears the screen, and redisplays the menu.

Strictly BATCH

Now try creating a menu of strictly batch files and placing this menu and all associated batch files in a subdirectory called \BATUTIL. Once this is accomplished, you can execute this menu with the change directory (CD) command to the \BATUTIL subdirectory and entering the command MAINMENU to run the MAINMENU.BAT file. You can also create a batch file called MENU.BAT that simply runs the MAINMENU.BAT file by including the one line command MAINMENU in that file. Another method to

Figure 8.4
Execution of BASIC Program that Accepts String Data

```
                        JANE's PC MAIN MENU
-----------------------------------------------------------------

                1 - WORDPROCESSING

                2 - SPREADSHEET

                3 - DATABASE


                0 - EXIT...Return to DOS

Enter the NUMBER of your choice... ? QUIT

*** ERROR ***  -->  You must enter a NUMBER between 0 and 3...

Press <ENTER> or <RETURN> key to try again...?
```

get the menu to appear automatically on the screen is to create an AUTOEXEC.BAT file in the root directory that automatically changes to the \BATUTIL directory and executes the MAINMENU.BAT file. An example of an AUTOEXEC.BAT file that performs the previously described operations whenever the computer is powered on or a system reset is performed is shown in Listing 8.16.

Listing 8.16 AUTOEXEC.BAT File

```
1: echo off
2: cls
3: path = \DOS;\BATUTIL
4: cd \batutil
5: MAINMENU
```

This AUTOEXEC.BAT file turns off the echo of commands to the screen in line 1. Line 2 clears the screen and line 3 sets the DOS search paths for commands and files to first look at the \DOS subdirectory and then look at the \BATUTIL subdirectory. Line 4 changes the current directory to the \BATUTIL subdirectory, and line 5 executes the MENU.BAT file to place the menu on the screen.

Listing 8.17 shows the contents of the MAINMENU.BAT file.

Listing 8.17 MAINMENU.BAT File

```
1:*echo off
2: cls
3: type mainmenu.txt
4: prompt Enter your selection please.. . .
```

Lines 1 and 2 turn echo of commands off and clear the screen, respectively. Line 3 types out a file called MAINMENU.TXT that contains the text that forms the menu. Line 4 changes the DOS prompt from the standard A> (C> on a hard disk) to the words `Enter your selection please....` Now, whenever the menu is displayed the user will see the `Enter your selection...` prompt instead of the C> prompt. Using the DOS PROMPT command to change the prompt has made the menu much more meaningful and easy for a novice to use. An experienced user will quickly recognize that the computer is still at DOS level and only the prompt has been changed. Any DOS command can still be given at this prompt.

Listing 8.18 shows the contents of the MAINMENU.TXT file, which in this example contains a menu for Tom. Notice that six options with numbers are available for use and an option for "U" for Utilities or "0" to exit to DOS can be chosen. Each of the options 1 through 6 is similar to the previous example of batch menus where the entry of a numeric results in batch files called 1.BAT, 2.BAT, 3.BAT, and so on, being invoked.

Listing 8.18 Text File MAINMENU.TXT

```
1:*
2:                          MAIN MENU FOR TOM
3: ================================================================
```

```
 4:
 5:
 6:      1 - SPREADSHEET                    U - Utilities
 7:
 8:      2 - WORD PROCESSING
 9:
10:      3 - DATABASE
11:
12:      4 - PROFS
13:
14:      5 - HARVARD GRAPHICS
15:
16:      6 - PAGEMAKER                      0 - Return to DOS
17:
18:
```

Listing 8.19 shows the 1.BAT file.

Listing 8.19 1.BAT File

```
1:*echo off
2: cls
3: cd \123
4: lotus
5: cls
6: cd \batutil
7: type mainmenu.txt
```

Here, lines 5, 6, and 7 are used to return to the main menu after the selected spreadsheet software is exited. Similarly, files 2.BAT through 6.BAT can be constructed, the only differences among them being in the location where the change of directory occurs (line 3) and in the command used to invoke the application program (line 4). But what about this option 0, which supposedly enables you to return to DOS? Listing 8.20 shows the O.BAT file.

Listing 8.20 O.BAT File

```
1:*echo off
2: cls
3: echo...Returning to DOS...
4: cd \
5: prompt $n$g
```

Lines 1 and 2 simply turn echo of the following commands to the screen off and clear the screen. Line 3 tells the user that the system is returning to DOS control, and line 4 changes the current directory to the root directory. Line 5 again uses the DOS PROMPT command to change the prompt back to the current disk drive and the greater

than symbol (C> in this case). Remember that you changed the prompt in MAINMENU.BAT and this simply changes the prompt back to its original value.

Listing 8.21 shows the U.BAT file, which brings up a Utility menu. Lines 1 and 2 simply turn off the echo of commands and clear the screen, and line 3 places the UTIL.TXT file, which contains the Utility menu, on the screen. The prompt has not been changed since MAINMENU.BAT, and you are still asked to `Enter your selection please...`

Listing 8.21 U.BAT File

```
1:*echo off
2: cls
3: type util.txt
```

Listing 8.22 shows the contents of the UTIL.TXT file used to create a utility menu. Note that each of the utilities in this menu uses letters for its execution. This is because these batch files are M.BAT, D.BAT, S.BAT, and so on. These are not explained here, but using the concepts provided in this book you could easily generate your own utility batch files for this menu. Then, to return to the MAINMENU.BAT file you would create a two-line R.BAT file whose contents would be ECHO OFF followed by MAINMENU on the second line.

Listing 8.22 UTIL.TXT File

```
 1:*
 2:                          UTILITY MENU
 3: ================================================================
 4:
 5:
 6:     M - Move files
 7:
 8:     D - Display file
 9:
10:     S - Sort Directory
11:
12:     P - Printer Control
13:
14:     L - Label generation
15:
16:     F - File locater                    R - Return to MAIN MENU
17:
18:
```

The Configuration File

The configuration file is a special type of batch file designed to hold commands to configure your computer system. This batch file must be given the name CONFIG.SYS

and can be created in the same manner as other batch files. That is, the CONFIG.SYS file can be created by the use of EDLIN, a word processor capable of creating ASCII files, or by using the DOS COPY command as shown here.

```
COPY CON: CONFIG.SYS
```

Similar to terminating input to any batch file when you are copying data from the console, you must press the F6 key or simultaneously press Ctrl+Z to terminate keyboard input to the CONFIG.SYS file.

Once you create a CONFIG.SYS file its execution will receive priority over all other files with the exception of the three system files—IBMBIO, IBMDOS, and COMMAND.COM. This priority of execution includes any AUTOEXEC.BAT file that may reside on your disk, because the commands contained in the CONFIG.SYS file can govern the operation of commands in the AUTOEXEC.BAT file. Figure 8.5 illustrates the relationship between the execution of the CONFIG.SYS file and an AUTOEXEC.BAT file with respect to the tasks performed by DOS during its initialization process. As indicated in this figure, each time you power-on your computer or perform a system reset, DOS searches the root directory of the drive it was initiated from for the file CONFIG.SYS. Then, if found, the file is executed.

Utilization

Commands that can be placed in the CONFIG.SYS file can be used to specify a country date and time format, specify the maximum number of drives that can be open at one time, specify a file or files that contain device drivers, and set other configuration-related parameters, such as whether DOS should check for Ctrl+Break. When shipped in the United States, DOS 3.3 normally contains a one-statement configuration file as illustrated.

```
C>TYPE CONFIG.SYS

COUNTRY=001
```

Here, the statement COUNTRY=001 specifies the time and date format for the United States. This can be easily altered by changing the country code to another three-digit code supported by DOS.

Under DOS 4.0, if you used the INSTALL program to place DOS on a fixed disk, your configuration file will appear similar to

```
BREAK=ON
BUFFERS=20
FILES=8
LASTDRIVE=E
SHELL=C:\DOS\COMMAND.COM /P /E:256
DEVICE=C: \DOS\ANSI.SYS
INSTALL=C:\DOS\FASTOPEN.EXE C:=(50,25)
```

The BREAK=ON statement results in DOS checking for the Ctrl+Break key sequence whenever the statement is entered. Otherwise, DOS only checks for it when I/O op-

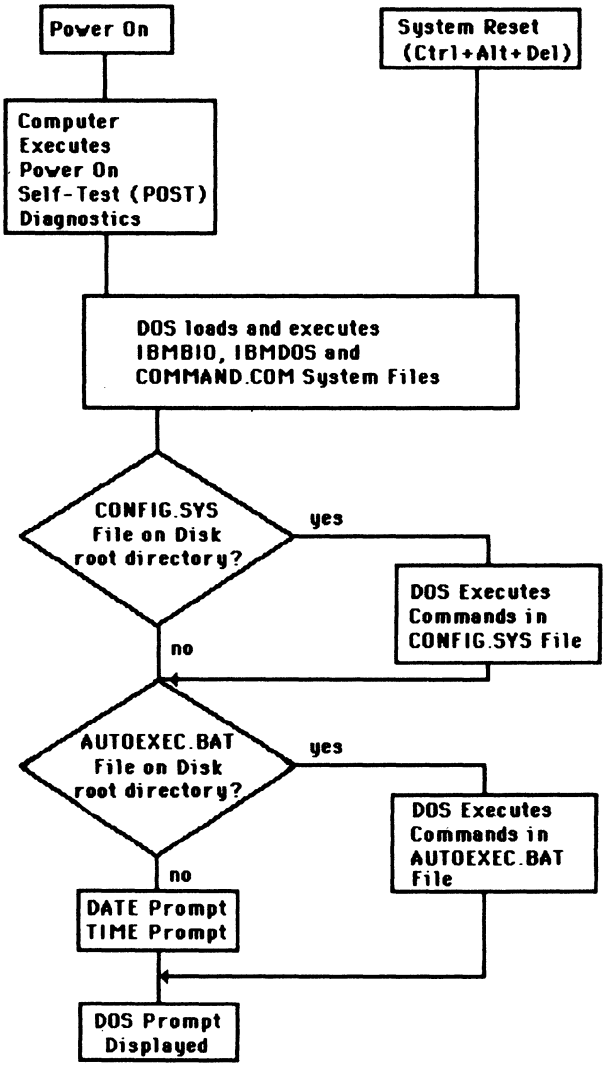


Figure 8.5
DOS Initialization Process

erations are performed. The BUFFERS=20 statement causes DOS to allocate 20 disk buffers in memory when it starts, whereas the FILES=8 statement permits up to 8 files to be open at the same time.

The LASTDRIVE=E sets the maximum number of drives you may access to five, including virtual disks created in memory. By changing this statement you can increase the maximum number of drives you can access to 16.

The SHELL=C:\DOS\COMMAND.COM statement loads and starts the DOS command processor, COMMAND.COM, located in the DOS subdirectory on drive C. The /P parameter installs COMMAND.COM as a permanent command processor, whereas the /E:256 optional parameter specifies an environment size of 256 bytes. This is the location

where DOS keeps track of the path or paths to search for commands not found in the current directory, the path DOS should use to reload the command processor when necessary, and other key information. You can display the current environment setting by entering the DOS SET command without parameters.

The `DEVICE=C:\DOS\ANSI.SYS` statement causes the ANSI.SYS file located in the DOS directory on drive C to be installed. The ANSI.SYS file contains an enhanced standard input and standard output device driver that replaces IBM's standard I/O support. ANSI is an acronym for the American National Standards Institute, and the use of ANSI.SYS provides a standard method of support for programming that performs such functions as reassigning the keyboard key, manipulating the cursor, and displaying color attributes.

The last statement in the CONFIG.SYS file normally created by the DOS 4.0 INSTALL program causes the FASTOPEN.EXE file to be installed when DOS is initialized. This program enables your computer to retrieve recently opened files faster than in earlier versions of DOS by storing file information in memory. The statement contained in the CONFIG.SYS file allocates 50 directory or file entry buffers in memory and 25 continuous space buffers for the files on drive C.

Configuration Commands

This section examines the operation and utilization of seven commonly used configuration file commands. Each of these commands can be included by itself or with any other configuration commands in a CONFIG.SYS file.

BREAK Command

The BREAK command enables or disables DOS checking for Ctrl+Break. If this command is not included in the CONFIG.SYS file, a default value of BREAK=OFF is assumed. The format of this command is

$$\text{BREAK=} \left[\left\{ \begin{matrix} \text{ON} \\ \text{OFF} \end{matrix} \right\} \right]$$

With BREAK=OFF, DOS checks for Ctrl+Break only during keyboard, printer, and asynchronous communications operations.

BUFFERS Command

The BUFFERS command tells your computer how much RAM to reserve for data buffers. DOS allocates up to 99 buffers, which are temporary storage areas of 512 bytes used as intermediate storage when data is accessed from disk. When requested to retrieve disk data, DOS sequentially examines the contents of each buffer until it finds the requested data. If it is not found, DOS performs a disk read, causing sectors on a track to be read into disk buffer storage and a search of the buffer contents to be repeated.

As a good rule of thumb, you should have at least one buffer for every open file. In actuality, the computation of an optimum number of buffers is almost impossible, because different application programs specify different BUFFER requirements. Normally,

you should set the BUFFER=*XX* command in your CONFIG.SYS file to the maximum value required by the application program you use most frequently. If you do not include a BUFFERS command in your CONFIG.SYS file, DOS uses a default of 15.

The format of the BUFFERS configuration command under DOS 3.3 is

```
BUFFERS=m
```

where *m* is a number between 1 and 99. Although it is difficult to set an optimum number of BUFFERS, for most applications a value set equal to the highest requirement of your set of application programs will suffice. Under DOS 4.0, BUFFERS can be set to 1 through 10,000, if expanded memory is available. In addition, BUFFERS now supports *look-ahead buffers,* which is the term used to represent the number of sectors your computer can read in advance of processing any input operation. Under DOS 4.0 the format of the BUFFERS command is

```
BUFFERS=m,n
```

where *n* is the number of look-ahead buffers that can be set from 1 through 8.

COUNTRY Command

As previously mentioned, the COUNTRY configuration command sets the date and time format of your computer. In addition, this command automatically sets the currency symbol and decimal separator for a country based on the three-digit country code specified in the command.

The format of the COUNTRY configuration command under DOS 3.3 is

```
COUNTRY=XXX
```

where *XXX* is a three-digit international country code for the telephone system of the country. Table 8.2 lists the country codes currently supported by DOS.

Under DOS 4.0 the COUNTRY configuration command was expanded to support the code page of the desired country. The code page is a table that translates numeric

Table 8.2
DOS 3.3 Supported Country Codes

Country	Country Code	Country	Country Code
Australia	061	Middle East	785
Belgium	032	Netherlands	031
Canadian-French	002	Norway	047
Denmark	045	Portugal	351
Finland	358	Spain	034
France	033	Sweden	046
Germany	049	Switzerland	041
Italy	039	United Kingdom	044
Israel	972	United States	001

information stored in your computer into the letters, symbols, and characters used in a particular language. The code page can be specified in COUNTRY configuration command by following the three-digit international country code by a comma and then entering the code page in the statement.

Table 8.3 lists the country codes and code pages supported by DOS 4.0. Refer to the DOS 4.0 manual for specific information on how you can switch among several pairs of code pages.

DEVICE Command

With the DEVICE configuration command you can specify the name of a file containing a device driver. Prior to the availability of the DEVICE statement, you normally had to separately load a .COM program to support special hardware to include mice and scanners, as well as virtual disks and other nonstandard devices. With the support of

Table 8.3
DOS 4.0 Supported Country Codes and Code Pages

Country	Country Code	Code Pages Supported
Arabic-speaking	785	864,[1] 850
Australia	061	437, 850
Belgium	032	850, 437
Canada (French-speaking)	002	863, 850
Denmark	045	850, 865
Finland	358	850, 437
France	033	437, 850
Germany	049	437, 850
Hebrew-speaking	972	862,[1] 850
Italy	039	437, 850
Japan	081	932,[1] 437
Korea	082	934,[2] 437
Latin America	003	437, 850
Netherlands	031	437, 850
Norway	047	850, 865
Portugal	351	850, 860
Simplified Chinese	086	936,[2] 437
Spain	034	437, 850
Sweden	046	437, 850
Switzerland	041	850, 437
Traditional Chinese	088	938,[2] 437
United Kingdom	044	437, 850
United States	001	437, 850

Notes:
1. This code page is supported only with a country supplement.
2. This code page is supported only with the Asian version of DOS 4.0 on Asian hardware.

a DEVICE statement, you can now tell the system which files to load and where they are located. This capability enables device drivers to be automatically invoked to coordinate the activities of DOS and nonstandard hardware. The format of this command is

```
DEVICE=[d:][path]filename[.ext]
```

Device drivers included on the DOS 3.3 diskette are ANSI.SYS, DRIVER.SYS, and VDISK.SYS. ANSI.SYS, as previously discussed, is an enhanced standard input and output device driver. DRIVER.SYS is a block device driver that permits disks to be referenced to a logical letter, and VDISK.SYS is a virtual disk device driver. Of the three device drivers supplied on DOS 3.3, VDISK.SYS is probably most popular, because it enables you to reserve a portion of RAM memory to be used as if it were a disk drive. Because a RAM disk provides far faster file access or data transfer than an electromechanical diskette or fixed disk, its use can be highly advantageous for programs that require numerous I/O operations.

Under DOS 4.0 the device drivers DISPLAY.SYS, DRIVER.SYS, and PRINTER.SYS were added. DISPLAY.SYS enables you to use code page switching on EGA displays and on the IBM PC Convertible LCD display. DRIVER.SYS allows DOS to assign a logical drive letter to any internal or external diskette drives you might add to your system. The third device driver added to DOS 4.0, PRINTER.SYS, enables you to use code page switching on some IBM ProPrinters and the IBM Quietwriter III printer.

To set up a virtual disk, you insert the DEVICE statement in your configuration file with the following format:

```
DEVICE=[d:][path]VDISK.SYS[size][sector][entries]
```

The optional drive letter and path denote the location where the VDISK.SYS file resides; `size` specifies the virtual disk size in K bytes, with 64K bytes used as a default value; `sector` is the sector size in bytes, with allowable sizes 128, 256, and 512, with a default value of 128; `entries` specifies the number of directory entries (files) that the virtual disk can contain. The range of the entries parameter is 2 to 512, with its default value being 64.

As an example of the use of the DEVICE configuration command, assume you want to set up a virtual disk of 360K bytes of RAM with a sector size of 512 bytes that can contain up to 64 files. To set this up, you enter the following statement in your configuration, assuming that the VDISK.SYS file resides in the root directory of the drive containing DOS. If the VDISK.SYS file is located on a different disk or directory, you must prefix its name with a drive designator and path to denote its location.

```
DEVICE=VDISK.SYS 360 512 64
```

Assuming you wish to set the time and date format to be used by DOS to that used for the United States and automatically create a virtual disk, you can use the COPY command to create a two-line CONFIG.SYS file. Its contents are

```
C>COPY CON: CONFIG.SYS
COUNTRY=001
DEVICE1=VDISK.SYS 360 512 64
```

```
^Z
        1 File(s) copied
C>
```

Note that similar to a batch file, the entry of data to the CONFIG.SYS file is terminated by pressing either the F6 key or the Ctrl+Z multikey combination.

Now that you created or modified an existing CONFIG.SYS file, you must reboot your system to execute the commands contained in that file. Thus, when you power-on your computer the next time or if you perform a system reset operation, the following message will be displayed, indicating that 360K bytes of memory were allocated to a virtual disk that was provided the drive designator D.

```
VDISK Version X.O virtual disk D:
      Buffer size:             360 KB
      Sector size:             512
Directory entries:      64

Current date is Sun 11-11-1988
Enter new date (mm-dd-yy):
```

Note that the virtual disk drive designator is based on the first unused disk on your system. Thus, if you already have two disk drives called A: and B:, the virtual disk is designated C:. Similarly, if you have drives A:, B:, and C:, the virtual disk is designated D:. DOS automatically supports up to five drives, through drive designator E. If you require the use of more than five drives, use the LASTDRIVE configuration command in the CONFIG.SYS file.

FILES Command

The FILES configuration command specifies the maximum number of files that an application program can have open at one time. The format of this command is

```
FILES=XXX
```

where *XXX* can have a value between 5 and 255. If the command is not included in a CONFIG.SYS file, a default value of 8 is used by DOS.

When DOS is initiated, it automatically opens five files—standard input, standard output, standard error, standard printer, and standard auxiliary device for each process. Thus, a default value of 8 is normally sufficient for most application programs. However, if your application program should return an error message indicating an insufficient number of files, you can use the FILES= configuration command to increase the maximum number of open files supported by DOS. To avoid an error message and having to change your configuration file, you should set FILES= to the maximum number of open files required by the application programs you use.

LASTDRIVE Command

The LASTDRIVE configuration command increases the maximum number of disk drives DOS will support. With this command you can add support for logical drives whose numbers exceed the number of physical drives installed in your system.

DOS permits the use of five disk drive names, A: through E:, in its default mode of operation. In order to have DOS recognize as valid a disk drive letter beyond E:, include the LASTDRIVE command in your CONFIG.SYS file. The format of this command is

```
LASTDRIVE=letter
```

where the letter represents the last valid disk drive you want to be able to use on your system. Any letter up to Z is permitted, permitting DOS to support up to 26 "disks" on your computer.

SHELL Command

The SHELL configuration command specifies any executable program to be loaded as a top-level shell processor. The top-level shell is a resulting user interface between the two system files, IBMBIO and IBMDOS, and an executing application. Normally this interface is COMMAND.COM, which is the DOS default shell.

The format of the SHELL command is

```
SHELL=[d:][path]COMMAND[.COM][d:][path][/C][/P][/D][E:n][/F]
```

The device identifier and path indicate where the initial copy of COMMAND.COM is located. The second drive and path are optional and are used to specify where COMMAND.COM should look when it needs to reload itself. If this location is not specified, COMMAND.COM reloads itself from the location where the initial copy of the program resides. This reloading is necessary because COMMAND.COM was designed to split itself into a resident and a transient partition to provide more memory for the execution of application programs. This enables large programs to overwrite the transient portion of COMMAND.COM, which is located in high memory. Then, after the program is terminated, the resident portion of COMMAND.COM attempts to reload the transient portion from its old location or another location specified in the SHELL statement. This also explains why, as an example, you may see the message `Insert disk with COMMAND.COM in drive X` displayed after an application program is terminated.

The /C (Command) parameter is used to cause a newly loaded COMMAND.COM to perform a specified internal command or to load and execute the program following the letter. The /P parameter tells COMMAND.COM it should make itself permanent. In addition, it also tells COMMAND.COM to locate, load, and execute your computer's AUTOEXEC.BAT file. The /D parameter, when used with the /P parameter, disables the automatic execution of the AUTOEXEC.BAT file when COMMAND.COM becomes permanent.

The /E parameter specifies the size in bytes to allocate to the environment. You can specify a range of numbers (*n*) from 160 to 32,768 (32K) bytes, with the default being 160 bytes.

The /F option causes the error handling code in COMMAND.COM to automatically fail any error. Thus, even though the familiar `Abort, Retry, Ignore` message will still be displayed, you do not have to enter any response to continue operations. This option should be used if you are operating your PS/2 via remote communications. Then, if you accidentally caused a critical error, you could continue operations without having to go to your computer to enter a keyboard reply to a critical error.

9 Advanced DOS

This chapter describes many of the advanced features of the disk operating system. Features covered in this chapter include I/O redirection, which encompasses the use of pipes and filters; file backup and restoration; and the DOS environment.

I/O Redirection

To understand the principle behind I/O redirection, a short review of standard computer I/O operations is warranted. This information provides a basis for understanding the principles of I/O redirection.

In the normal or default state of operation of a PS/2, the operating system expects to receive input from the keyboard while output is directed to the video display. These two devices—keyboard and video display—are also known as DOS's standard I/O devices. Unless specified otherwise, DOS commands result in standard I/O operations. Thus, a directory listing normally is displayed on your screen, whereas the command DIR that results in the display is normally input from your keyboard. Although standard I/O is the default method by which I/O is serviced and will suffice for most applications, on occasion you will want to redirect I/O operations. To accomplish this, you can use the greater than (>) and less than (<) characters in a DOS command line.

Input Redirection

To specify input redirection, use the less than symbol (<) between a command and a reserved name or file specification. Thus, the format for input redirection is

DOS *COMMAND*< { *reserved name* / *file specification* }

where the braces indicate that a choice of one of the enclosed items is to be made.

One of the more common uses of input redirection is to change the execution requirements of programs. As an example of this, consider a program named PAYROLL whose execution normally requires the computer user to sit at the terminal and type a series of keyboard entries when the program executes. Using input redirection, you can first use the EDLIN text editor included on the DOS diskette or a word processor to create an ASCII file containing the keyboard entries required to execute the program.

Assuming that the name of the file containing the keyboard entries is JAN88.DAT, the command

```
PAYROLL<JAN88.DAT
```

causes the PAYROLL program to execute with input to the program occurring from the file named JAN88.DAT.

One of the major advantages of input redirection is its automation of program execution. A secondary advantage that may be just as important with respect to productivity is the ability to store program input data in files. Then, if there are minor changes to the input data between successive program executions, a word processor or text editor can expedite making the required changes.

Output Redirection

Output redirection is similar to input redirection in that both can be used with reserved names or file specifications. The format of output redirection is

DOS *COMMAND* { > / >> } { *reserved name* / *file specification* }

When output is redirected to a file, DOS first checks whether the file exists. If it does, DOS overwrites the existing file with the output *unless* two greater than symbols (>>) are included in the output redirection. In this case, DOS appends the requested output to the end of a previously created file. If the file does not exist, DOS creates and automatically saves the requested file.

To understand the benefits of output redirection, consider the following DOS command line.

```
DIR>LPT1:
```

When this command line is executed, DOS routes the directory listing to the first parallel printer port, providing a mechanism to automatically obtain a hard copy of the directory.

Returning to the PAYROLL program discussed under input redirection, suppose the output of the program directed to the screen contains `successful` or `error` messages for five modules. In addition, assume that the messages scroll off the screen as the next module executes. Because the creation and use of an input file to respond to the program's prompts might still require the computer operator to read each screen, the operation is not really automated. In this instance, the computer user might consider combining input and output redirection, as shown in this command:

```
PAYROLL<JAN88.DAT>SCREEN.PIC
```

In this example the payroll program receives input from the file JAN88.DAT and sends screen output to the file SCREEN.PIC. Later, the operator can use the DOS TYPE command to examine the screen information previously generated by the program.

Pipes and Filters

Piping enables you to chain DOS commands and programs with the automatic redirection of standard I/O, permitting the screen output of one command to be used as the keyboard input to another command or program.

The special symbol (¦) denotes a pipe and serves as a delimiter between the output of one command and the input to the second command. The second command, which accepts data from a standard input device, modifies the data and then outputs the results to a standard output device, is commonly known as a *filter*. Currently, DOS includes three filter commands—SORT, MORE, and FIND.

SORT (External)

The SORT command reads data from a specified input device, sorts the data, and then writes it to the standard or a specified output device. The format of this command is

```
[d:][path]SORT[/R][/+n]
```

Here the optional /R switch causes the sort to be performed in reverse order, whereas the /+n switch causes the sort to start with column n. If the second option is not specified, the sort or reverse sort starts in column 1.

The format of the piping of standard I/O is

```
DOS COMMAND ¦ DOS COMMAND { > | >> | < } { reserved name | file specification }
```

As an example of piping using the SORT command, consider the following command line entry.

```
DIR C:¦SORT>LPT1:
```

In this example the video output that normally results from the directory (DIR) command is first piped into the SORT command, which sorts the directory and then directs the output of the sorted directory to the first parallel printer port.

MORE (External)

The MORE command is a filter that displays one screen of 24 lines of data and the message -More-, then pauses. By pressing any key you can display the next screenful of data. The format of this command is

```
[d:][path]MORE
```

As an example of the use of the MORE filter, consider the following command line entry.

```
MORE<SORT/R<MEMO.DAT
```

The preceding command line causes the file MEMO.DAT to be sorted in reverse order and then displayed one screen at a time. Note that the SORT command sorts data

according to the ASCII values of the characters it encounters. This means that when you use MORE with a data file, the file must be in ASCII format. In addition, because uppercase letters have a lower ASCII value than lowercase letters, the results of a sort operation can be unexpected. For example, PREPARATION will appear before Parameter. For this reason, good sort utilities (which the DOS SORT command is *not*) internally convert the sort keys to a single case before comparing them, so that the sorted output is in proper alphabetical order. SORT is also slower than a sick tortoise—don't try to sort more than 50 items with it unless your patience is infinite!

FIND (External)

The third filter in DOS is the FIND command. This command can be used to search an ASCII file for the occurrence or nonoccurrence of a string included in the command. The format of the command is

```
FIND[/V][/C][/N]"string"[file specification]
```

The /V switch causes FIND to display all lines that do *not* contain the specified string. The /C switch causes FIND to count the number of lines that contain the specified string, whereas the /N option causes the command to include the line number for each displayed line.

The FIND command provides a valuable mechanism for constructing a sequence of commands to perform database-related operations. To better understand the utility of this command, assume you have an ASCII file named PHONE.DAT whose record format is illustrated in Figure 9.1.

Suppose the file was created and is maintained with the aid of a word-processing program at a central office within an organization. Other persons in the organization might be interested in obtaining a telephone directory of persons in their department. To do this, they would use the FIND filter to locate all records in the file that contain the string associated with their department code. If the department code of interest is A106, all the records containing that string could be printed using the following command line.

```
FIND "A106" PHONE.DAT>LPT1:
```

If for some reason you desired a directory listing based on the department code for the entire organization, you could use the SORT command with the FIND command in one line to print it. Because the department code commences in column 37 of each record, you must signify this in the SORT command using the /+n option, with n assigned the value 37. Thus, you could use the FIND and SORT commands, piping the output

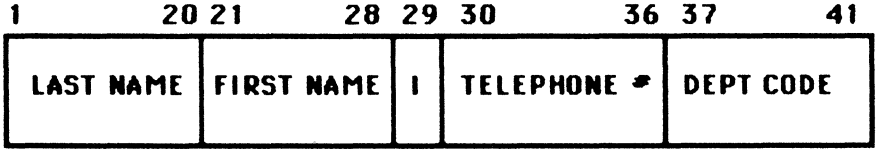


Figure 9.1 Telephone Record Format

of FIND to SORT and directing the output of the SORT command to the printer, as shown here.

```
FIND "A106" PHONE.DAT¦SORT/+37>LPT1:
```

File Backup and Restoration

Three of the most important, but frequently forgotten, DOS commands for fixed disk operations are BACKUP, XCOPY, and RESTORE. Although each of these commands can be used to back up files from one diskette to another, their primary use is in providing computer users with a mechanism to back up the contents of a fixed disk to diskette storage and to restore files from diskette storage to a fixed disk. By performing backup operations on a regularly scheduled basis, you can minimize the effect of a hard disk head crash that could conceivably wipe out tens to hundreds of millions of bytes of storage, as well as countless hours of prior effort.

BACKUP (External)

The BACKUP command duplicates one or more files from one type of disk to another, as long as the drive specifiers in the command are not duplicated. The format of this external DOS command is

```
[d:][path]BACKUP
d:[path][filename[.ext]]d:[/S][/M][/A][/D:mm-dd-yy]
[/T:hh:mm:ss][/F][/L[:[d:][path][filename[.ext]]]]
```

The drive specifier following the command keyword denotes the source drive, which contains one or more files to be backed up. The path, filename, and extension are used to identify the location of the files to be backed up, whereas the optional parameters further define the characteristics and attributes to be used during the backup process. Table 9.1 lists the operational result of each of the BACKUP command parameters.

Table 9.1 BACKUP Command Parameters

Parameter	Operational Result
/S	Backs up subdirectory files in addition to the files in the specified or current directory.
/M	Backs up files modified since the last BACKUP operation.
/A	Appends files to files already present on the backup disk.
/D	Backs up files modified on or after the specified date.
/T	Backs up files modified on or after the specified time on the specified date.
/F	Formats the target diskette if it is not.
/L	Creates a log file. If a filename is not specified, a default of BACKUP.LOG will be created in the root directory of the source drive.

By including one or more optional parameters in the command line, you can precisely control the files to be backed up based on their directory location and date and/or time of creation. In addition, when you use diskette storage for backup under DOS 3.3, it is advisable to include the /F parameter in the command line. Otherwise, if you run out of formatted diskettes you will have to use a different computer to format additional disks. To do so on your computer, you must terminate the BACKUP operation and restart the operation when you have the correct number of formatted diskettes.

Under DOS 4.0 the target diskette is automatically formatted if it was not already formatted by the FORMAT command. Due to this, the /F parameter has been eliminated from the BACKUP command under DOS 4.0.

Even though using the DOS Shell simplifies the BACKUP and RESTORE operations, you should be familiar with the optional parameters of these commands as well as other DOS commands. Figure 9.2 illustrates the Backup Utility pop-up box that is displayed when you select the `Backup Fixed Disk` action from the DOS Utilities screen. Note that the default parameter setting is

```
c:\*.* a: /s
```

Here `c:\*.*` denotes all files in the root directory of drive C, whereas `a:` indicates those files should be backed up onto a diskette in drive A. Finally, the /S parameter causes the backup of subdirectory files in addition to the files in the root directory, in effect causing the entire contents of drive C to be backed up.

Figure 9.2 Backup Utility Pop-Up Box

Because a complete backup of a fixed disk can take a significant amount of time, you will probably prefer to use other parameters after the first backup operation is performed. Thus, becoming familiar with the use of command parameters can significantly increase your efficiency in using your computer.

The number of diskettes required for backup operations depends on your backup requirements and your diskette storage capability. If your personal computer has a 20M byte fixed disk that is half full, to back up the 10M bytes of data requires fourteen 720K byte formatted diskettes on a PS/2 Model 25 or 30 computer; whereas only seven 1.44M byte formatted diskettes are required if your PS/2 uses high-capacity 3½-inch diskettes. Because it is very difficult to estimate the amount of data that changed from the time of a previous backup operation, a good rule of thumb is to always have a blank box of diskettes available, as well as to use the /F parameter in the BACKUP command line if you are using DOS 3.3.

Note that BACKUP is not the same as COPY and should not be used to duplicate files. This is because the COPY command creates an exact duplicate of a file or group of files, whereas the BACKUP command chains files together and inserts control information that precludes its use unless restored to its original format by the use of the RESTORE command.

When you use the BACKUP command with a diskette drive as the target drive, you are prompted to insert a new diskette after the currently inserted diskette is filled. You should always label each diskette to include recording its date and diskette number, because the RESTORE operation requires backup diskettes to be inserted in the order they were created. In addition to prompting you to insert an appropriate diskette, BACKUP issues a warning that any files previously stored in the root directory of the disk used for backup will be erased. This warning arises from the BACKUP technique of creating two files in the root directory on a target diskette—BACKUP.XXX and CONTROL.XXX. The BACKUP.XXX file contains all the files chained together, whereas the CONTROL.XXX file contains such control information as path names and filenames. The following illustrates the use of the BACKUP command to back up all files on drive C in the HIJACK subdirectory onto a diskette in drive A. Note that once the command is in operation, the name of each file backed up is displayed as it is transferred to the backup media.

```
C>BACKUP \HIJACK A:

Insert backup diskette 01 in drive A:

Warning! Files in the target drive
A:\ root directory will be erased
Strike any key when ready

*** Backing up files to drive A: ***
Diskette Number: 01

\HIJACK\UPDATE
\HIJACK\HIJACK.EXE
```

```
\HIJACK\CONVERT.EXE
\HIJACK\CAPTURE.EXE
\HIJACK\EGA.EXE
\HIJACK\TOSHIBA.EXE
\HIJACK\ATT.EXE
```

To help you achieve a better understanding of the use of the BACKUP command, Table 9.2 illustrates three command line entries and their operational results. As indicated by the first example in Table 9.2, you can use global characters in the BACKUP command to specify file sets.

RESTORE (External)

The RESTORE command can be considered to be the complement of the BACKUP command, because it reinstates on the fixed disk one or more files that were previously backed up onto another medium. The format of this command is

```
[d:][path]RESTORE d:[d:][path]filename[.ext][/S][/P][/B:mm-dd-yy]
[/A:mm-dd-yy][/M][/N]:[/L:time][/E:time]
```

The first drive identifier following the command keyword denotes the drive that contains one or more files previously produced by the use of the BACKUP command. The option drive identifier, as well as the following path, filename, and extension, specify the location where you want the files restored and what files are to be restored. Similar to the BACKUP command, you can use global characters in the filename and extension fields to specify a file set.

The optional parameters following the filename extension govern the conditions under which file restoration will occur. Table 9.3 lists the operational result of each of the RESTORE command parameters.

Table 9.2
BACKUP Command Examples

BACKUP Command	Operational Result
C>BACKUP *.* A:/S	Uses the BACKUP command file on the default drive (drive C) in the current directory to back up all files on that drive to the diskette in drive A.
C>BACKUP *.DAT A:/D:9-11-88	Uses the BACKUP command file on the default drive (drive C) in the current directory to back up all files in the current directory with the extension .DAT on drive C modified on or after 9-11-88 to drive A.
A>C:BACKUP C:\ A:/F	Uses the BACKUP command file on drive C in the current directory to back up all files from the root directory on drive C to drive A. If a diskette inserted in drive A is not formatted, BACKUP formats it prior to backing up any files.

Table 9.3
RESTORE Command Parameters

Parameter	Operational Result
/S	Restores all files in the specified directory and subdirectories under the specified directory.
/P	Causes RESTORE to prompt you prior to restoring files that have changed since the last backup or that are marked read only.
/B:mm-dd-yy	Restores all files modified on or before the specified date.
/A:mm-dd-yy	Restores all files modified on or after the specified date.
/M	Restores all files modified or deleted since they were backed up.
/N	Restores files that no longer exist on the target drive.
/L:time	Restores only those files modified at or after the specified time.
/E:time	Restores only those files modified at or earlier than the specified time.

Table 9.4
RESTORE Command Examples

RESTORE Command	Operational Result
C>RESTORE A: \HIJACK*.PIX	Uses the RESTORE command file on the default drive (drive C) in the current directory to restore files whose extension is .PIX in the subdirectory HIJACK on drive A to drive C.
C>RESTORE A: C:*.* /S	Uses the RESTORE command file on the default drive (drive C) to restore all files on the backup diskette(s) to drive C.
A>C:RESTORE A: C:*.* /S/N	Uses the RESTORE command file on drive C in the current directory to restore all files on the backup diskette(s) that no longer exist on drive C.

If you want to restore the files previously backed up in the sample backup procedure, enter the command

```
C>RESTORE A: C:\HIJACK
```

The preceding command restores all files in the directory HIJACK on drive A to the fixed disk, drive C. In using the RESTORE command you must remember that the first drive identifier indicates the source, the second drive identifier and path indicate the target, and the following file specification indicates what files from the source you want restored. Thus, the RESTORE command requires you to enter parameters in the sequence source, target, source. The following illustrates the use of the RESTORE command to recover the files in the directory HIJACK on drive A that were previously backed up. Table 9.4 illustrates the operational result from three RESTORE command line entries.

```
C>RESTORE A: C:\HIJACK

Insert backup diskette 01 in drive A:
Strike any key when ready

*** Files were backed up 06/30/1988 ***

*** Restoring files from drive A: ***
Diskette: 01
\HIJACK\UPDATE
\HIJACK\HIJACK.EXE
\HIJACK\CONVERT.EXE
\HIJACK\CAPTURE.EXE
\HIJACK\EGA.EXE
\HIJACK\TOSHIBA.EXE
\HIJACK\ATT.EXE
\HIJACK\HERC.EXE
\HIJACK\MONO.EXE
```

XCOPY (External)

Unlike the BACKUP command, which creates files you must RESTORE prior to usage, you can use files created by the XCOPY command without executing an intermediate command. XCOPY can be used in a backup procedure by specifying the /M optional parameter in the command line. This option turns off the archive bit of the source file, which informs both subsequent XCOPY and BACKUP commands that the file was not created or modified since the last use of either command. Due to this, you can intermix the use of XCOPY and BACKUP to satisfy your specific requirements. Refer to Chapter 5 for specific examples covering the use of the XCOPY command.

10 / Using Your Printer

This chapter presents a variety of techniques you can consider to obtain better performance by printers you attach to your computer.

The first part of the chapter focuses on printer control codes. *Printer control codes* are individual characters or strings of characters transmitted to a printer to enable or disable a printing feature. Although you can choose among a wide variety of printers, a subset of control codes are essentially used as de facto standards to obtain predefined printing operations. You will be introduced in this chapter to these predefined control codes, as well as a few differences between popular printers. You will be aware of how to check your printer's manual to ensure that programs you develop or purchase will work correctly with your printer(s).

After the examination of printer control codes, a section investigates the use of a programming language, as well as DOS commands, to issue control codes to a printer. This discussion is followed by the creation of a program that swaps printer ports, a useful utility if you have both a letter-quality printer and a dot-matrix printer attached to your system. Concluding this chapter is a discussion on the use of the DOS PRINT command. This command provides a limited multitasking capability, because it permits printing of files to occur concurrently with other operations. Thus, some PS/2 users may avoid switching to a different operating system to obtain a multitasking capability if the users simply want to perform work while the printer attached to the computer is in use.

Printer Control Codes

Printer control codes are individual characters or strings of characters that are transmitted to a printer to enable or disable predefined printing features. To differentiate between data to be printed and control codes, printers either contain circuitry that examines American Standard Code for Information Interchange (ASCII) characters received or a microprocessor and read only memory (ROM) code that performs a similar function.

Among the more common printer control codes are several ASCII characters whose decimal values are less than 32. When the ASCII character set was defined, all characters with a decimal value of 32 or less were assigned to nonprintable functions. These functions included ringing a terminal's bell (ASCII 7), performing a backspace (ASCII 8), tabbing horizontally (ASCII 9), and so on. Although the ASCII character set was

defined decades prior to the manufacture of printers for use with personal computers, many of the nonprintable control codes of the character set are used by modern printers. Table 10.1 lists the nonprintable ASCII control codes and the function they perform when received by a printer.

As you read the entries in Table 10.1, you may find several items that warrant an explanation. First, the ASCII Code column lists the mnemonics for each ASCII character. Although some are self-explanatory—such as LF for line feed—others—such as SO and SI and DC1 through DC4—are probably not recognizable unless you are familiar with the ASCII character set. SO is the mnemonic for Shift Out, which in communications is used to select a new character set. Similarly, SI is the mnemonic for Shift In, which in communications restores the original character set used. When dot-matrix printers were developed, manufacturers selected SO as the control code to initiate one line of double-width printing, whereas SI was selected as the control code to select condensed printing.

The DC1 through DC4 ASCII control characters are mnemonics for Device Control 1 through Device Control 4. When the ASCII character set was originally defined, most terminals were electromechanical teleprinters that could be interfaced to paper-tape readers and punches, cassettes, and other off-line storage devices. Originally, DC1 through DC4 control codes were designed to activate such devices. With the development of more modern printers, the DC1 through DC4 control codes were reserved to perform the functions indicated in Table 10.1.

Although most of the ASCII control codes listed in Table 10.1 will work with all types of printers, there are some exceptions. Among the major exceptions are the control

Table 10.1
Nonprintable ASCII Control Codes

ASCII Code	Decimal Value	Function
NUL	00	Null character
BEL	07	Sound beeper
BS	08	Backspace
HT	09	Horizontal Tab
LF	10	Line Feed
VT	11	Vertical Tab
FF	12	Form Feed
CR	13	Carriage Return
SO	14	Select Double-Width Printing (one line)
SI	15	Select Condensed Printing
DC1	17	Select Printer
DC2	18	Cancel Condensed Mode
DC3	19	Deselect Printer
DC4	20	Cancel Double-Width Printing (one line)
CAN	24	Cancel
ESC	27	Escape
SP	32	Space

codes that select and cancel double-width and condensed mode printing. Double-width printing results in the printing of characters twice the width of a normal character, whereas condensed-mode printing causes characters to be printed at about 60 percent of their normal width. Both of these print modes are only applicable to dot-matrix printers that have a printhead containing pins that are activated in correspondence to the print mode selected. If you have a letter-quality printer, its print pattern is fixed. Thus, the transmission of control codes to vary the print mode is either ignored by a letter-quality printer or produces unexpected results.

Because IBM PS/2 computers use extended ASCII, you can add 128 to each of the decimal values listed in Table 10.1 to perform the listed functions. This occurs because the eighth bit position of an 8-bit extended ASCII character is ignored by most ASCII printers that support characters whose values range from 0 through 127. Thus, ASCII 07 or ASCII 135 could, as an example, be sent to a printer to sound its beeper.

Escape Sequences

Because most modern dot-matrix printers are built to perform a large number of functions, a logical question is how those functions can be activated if ASCII 33 through 127 are printable characters that represent data. The answer to this question is the use of escape sequences that printers are constructed to interpret.

An escape sequence begins with the ASCII ESC character, whose decimal value is 27. The ESC character is then followed by one or two alphanumeric character codes that define the print function to perform. To illustrate the similarities and differences that can exist among escape codes recognized by printers, the IBM ProPrinter and Epson LQ-850/1050 printers were examined. Table 10.2 lists common escape code sequences recognized by each printer. As you read the entries in Table 10.2, note that although the operations performed by each printer are similar for many escape code sequences, there are also differences. Most of these differences are related to the physical construction of the printers, such as their tolerance for incrementing line spacing. Other differences between these two printers, as well as most dot-matrix printers, concern the character sets supported by each printer. Some printers have several built-in character sets, also known as *fonts*. Other printers are designed to accept one or more font cartridges that contain ROM, which defines how each character will be printed. When you use a printer that has two or more built-in character sets or that supports font cartridges, you must either send an appropriate code to the printer to select the desired character set or press a button on the printer's control panel.

Due to the subtle, but important, differences between escape code sequences printers are built to recognize, several software firms that develop printer drivers sold to word-processing developers have found a significant market for their products. Although most word-processing programs support a wide range of printers, they do not support *all* printers. Thus, by knowing how to send control codes to your printer you may be able to use your printer to perform certain functions the word processor or other application program you are using does not support.

Now that you have an understanding of printer control codes and the use of escape code sequences, apply that knowledge to controlling your printer. First, examine the

Table 10.2
Common Escape Code Sequences

Code	Decimal Values	Description	Printer
ESC 0	27 48	Select 1/8-inch line spacing	Both
ESC 1	27 49	Select 7/72-inch line spacing	IBM
ESC 2	27 50	Start text line spacing set by ESC A	IBM
ESC 2	27 50	Select 1/6-inch line spacing	Epson
ESC 3 n	27 51 n	Graphics line spacing (n/216 inch)	IBM
ESC 3 n	27 51 n	Graphics line spacing (n/180 inch)	Epson
ESC 4	27 52	Set top of form	IBM
ESC 4	27 52	Select italic mode	Epson
ESC 5	n 27 53 n	Automatic line feed (1=on,0=off)	IBM
ESC 5	27 53	Cancel italic mode	Epson
ESC 6	27 54	Select character set 2	IBM
ESC 6	27 54	Enable printable characters	Epson
ESC 7	27 55	Select character set 1	IBM
ESC 7	27 55	Enable upper control codes	Epson
ESC :	27 58 12	CPI printing	IBM
ESC :	27 58	Copy ROM to RAM	Epson
ESC A n	27 65 n	Select n/72-inch line spacing	IBM
ESC A n	27 65 n	Select n/60-inch line spacing	Epson
ESC B n..n	27 66 n..n	Set vertical tabs n..n	Both
ESC C n	27 67 n	Set form length (n lines)	Both
ESC C 0 n	27 67 0 n	Set form length (n inches)	Both
ESC D n..n	27 68 n..n	Set horizontal tabs n..n	Both
ESC E	27 69	Start emphasized printing	Both
ESC F	27 70	Cancel emphasized printing	Both
ESC G	27 71	Start double-strike printing	Both
ESC H	27 72	Cancel double-strike printing	Both

use of a statement and a function in BASIC that enables you to create a program to control the functions of your printer. Then examine how you can control the printer through DOS.

Control via BASIC

Because BASIC is provided on each DOS diskette, it's convenient for you to examine the control of the operation on your PS/2's printer using that language. Here the key to controlling printer functions is the BASIC LPRINT statement and the CHR$ function.

LPRINT Statement

The LPRINT statement can print information under program control. The format of this statement is

LPRINT [*list of expressions*] $\left\{\begin{matrix}[;]\\[,]\end{matrix}\right\}$

The expressions listed in a LPRINT statement may be numeric or string; however, string constants must be enclosed in quotation marks or they are considered to be variables.

If a semicolon (;) is used as an item separator, the value of the next item is printed directly after the previous item. If a comma is used as an item separator, the value of each item is placed into predefined positions called *print zones.* IBM BASIC is similar to most BASIC implementations in that a print line is divided into six print zones. If more than six items are in the list of expressions and you are using commas as separators, the values of the seventh through twelfth items will be printed on the next line in zones 1 through 6, and so on. If you terminate an LPRINT statement without a comma or semicolon, a carriage return and line feed are automatically generated by BASIC. Then, the next LPRINT statement prints the next chunk of information on a new line. If you include a comma or semicolon at the end of the LPRINT statement, the carriage return and line feed are suppressed, and the next LPRINT statement prints information on the same line.

CHR$ Function

The CHR$ function can be used in an LPRINT statement to send the ASCII character represented by the decimal value contained in the function to the printer. When you use it in an LPRINT statement, the format of the CHR$ function is

CHR$(*expression*)

The expression can be a decimal number or a mathematical expression that BASIC will evaluate. Because the IBM PS/2 uses an extended ASCII character set in which each character is represented by 8 bits, the expression can range in value from 0 to 255.

Now that you have reviewed printer control codes and the BASIC LPRINT statement and CHR$ function, try actually creating some small program segments to demonstrate how to operate some printer functions under program control. Suppose you have a dot-matrix printer that is capable of near-letter-quality (NLQ) printing but that cannot be placed into that printing mode by a switch or button on the control panel of the printer. If your word processor does not support the NLQ printing mode of your printer you can still use its NLQ printing capability by creating a small BASIC program to set that print mode. Then you can use this program each time you power-on your computer to set your printer to its NLQ print mode.

If you have an IBM ProPrinter I, that printer's NLQ print mode is also called double-strike printing. Using Table 10.1 you can determine that the ProPrinter requires ESC G (decimal 27 71) to switch to its NLQ print mode. You could send this code sequence

to your printer by executing the following LPRINT statement, which uses line number 100 for reference because each BASIC line requires a line number.

```
100 LPRINT CHR$(27) CHR$(71)
```

To print in NLQ print mode on a regular basis, you can avoid repeatedly entering and executing the LPRINT statement by creating a small file. If you want to execute a non-BASIC application after you set the NLQ print mode of your printer, you can add a second line, so your program looks like

```
100 LPRINT CHR$(27) CHR$(71)
110 SYSTEM
```

Here the BASIC SYSTEM command causes the program to exit back to DOS. Now that you have created the two-line BASIC program, assume you saved it, using the name NLQ.BAS. By typing SYSTEM without a line number you can exit BASIC and return to DOS. Now you can create a short batch file that automatically executes the previously created NLQ.BAS program. The creation of a batch file, named NLQ.BAS here, consists of

```
COPY CON: NLQ.BAT
BASICA NLQ.BAS
^Z
```

The batch file simply executes advanced BASIC (BASICA) using the previously created NLQ.BAS program file as input. The BASIC NLQ program file then sends an ESC G printer control sequence to your printer, placing it into its NLQ printing mode of operation. Once you have created the NLQ.BAS and NLQ.BAT files you can simply enter the command NLQ at the DOS prompt level, assuming your current directory is the directory that contains the NLQ.BAT file. Thus, each time you use your printer's NLQ print mode, you can power-on your printer and computer and enter the batch command NLQ.

Expanded Printer Control

On occasion you may want the ability to control specific features of your printer. You can develop a BASIC program that provides an easy mechanism to set and reset desired printer features.

This section provides an example of the use of the BASIC language to control a printer's mode of operation. You will see how to create a small program to control double-width, condensed, emphasized, and double-strike printing on both an IBM ProPrinter and an Epson LQ-850/1050 printer.

Based on the control codes listed in Tables 10.1 and 10.2, there are four 1-character control codes and four 2-character control code sequences that must be transmitted to either printer to enable and disable the four desired printer modes of operation. Table 10.3 summarizes the control codes as BASIC CHR$ functions that must be incorporated into LPRINT statements. Listing 10.1 illustrates the contents of a short BASIC program that controls the four printing modes previously discussed.

Table 10.3
CHR$ Functions for Selected Printer Modes

Printer Mode of Operation	BASIC CHR$ Function
Start double width	CHR$(14)
Cancel double width	CHR$(20)
Start condensed	CHR$(15)
Cancel condensed	CHR$(18)
Start emphasized	CHR$(27) CHR$(69)
Cancel emphasized	CHR$(27) CHR$(70)
Start double strike	CHR$(27) CHR$(71)
Cancel double strike	CHR$(27) CHR$(72)

Listing 10.1 BASIC Program to Control Four Printing Modes

```
100 CLS
110 PRINT "*** PRINTER SETUP UTILITY ***"
120 PRINT
130 PRINT " 1-Start double width"
140 PRINT " 2-Cancel double width"
150 PRINT " 3-Start condensed"
160 PRINT " 4-Cancel condensed"
170 PRINT " 5-Start emphasized"
180 PRINT " 6-Cancel emphasized"
190 PRINT " 7-Start double strike"
200 PRINT " 8-Cancel double strike"
210 PRINT " 9-Exit to DOS"
220 PRINT
230 INPUT "Enter selection 1 to 9";X
240 ON X GOTO 310,320,330,340,350,360,370,380,390
310 LPRINT CHR$(14);
315 GOTO 100
320 LPRINT CHR$(20);
325 GOTO 100
330 LPRINT CHR$(15);
335 GOTO 100
340 LPRINT CHR$(18);
345 GOTO 100
350 LPRINT CHR$(27) CHR$(69);
355 GOTO 100
360 LPRINT CHR$(27) CHR$(70);
365 GOTO 100
370 LPRINT CHR$(27) CHR$(71);
375 GOTO 100
380 LPRINT CHR$(27) CHR$(72);
```

```
385 GOTO 100
390 SYSTEM
```

Line 100 of the program clears the display screen. Lines 110 through 220 display the selection menu and the numeric code the user should enter to start or cancel one of the four printing modes or to exit BASIC to DOS. The INPUT statement in line 230 displays the message `Enter selection 1 to 9` and then assigns the numeric value entered to the variable X.

In line 240 the ON-GOTO conditional branching statement causes execution to branch to the appropriate line in the statement, based on the value of the X variable. That is, if X has a value of 1, a branch to line 310 occurs; if X has a value of 2, a branch to line 320 occurs; and so on. Each branch location uses an LPRINT statement with the CHR$ function to transmit the appropriate printer control code to the printer.

Although Listing 10.1 only controls four printing modes, you can easily add additional printing modes following the same logic incorporated in that program. You may have other types of printers than the IBM ProPrinter or the Epson LQ-850/1050 printer, and if so, you should review your printer manual to determine the control codes required to perform any printer control functions you may require. Then you can develop a program similar to the one listed in Listing 10.1 to control the printer modes of operation you require.

Printer Control Through DOS

If your only requirement is to set a few printer modes of operation, you can consider the direct use of DOS, eliminating the previously described method that incorporated a BASIC language program. Here the key to the use of DOS is the ability of the operating system to directly send printer control characters to your printer.

You can send any printer control character from DOS to your printer by pressing the ALT key and digits from the numeric keypad. Simply press and hold down the ALT key while you type the numeric value of the ASCII code for the character you desire. As an example, the decimal value of the escape character is 027. Thus, you would enter an escape character into a batch file by pressing and holding the ALT key while you enter the decimal value 027 from the numeric keypad.

If you attempt to enter the escape code whose ASCII value is 027, you will note that DOS will interpret that character as a request to stop accepting data. In effect, ASCII 027 is interpreted literally as a request to escape from performing a current operation. Thus, DOS not only stops accepting data but also terminates the current line entry and resets itself to accept a new command line entry. Due to this escape interpretation, you cannot directly enter a conventional escape character that has an ASCII value of 027. Fortunately, you can fool DOS into accepting a disguised escape code. This is accomplished by adding the value of 128 to 27 for a new value of ASCII 155. Here the addition of decimal 128 is equivalent to changing the high-order eighth bit of the ASCII escape character from a zero to a one. If you enter ASCII 155, DOS interprets the character as a nonescape character code. Now that you have a method of entering a disguised escape character, what is its effect when it is sent to the printer?

Printer Operations

Most printers only use the first seven bits of a character when they operate in a normal ASCII print mode where the character set is assumed to range in value from decimal 0 through decimal 127. When operating in this print mode the printer ignores the setting of the eighth bit, resulting in ASCII 27 and ASCII 155 both being interpreted as the escape character. Thus, the ALT+155 key combination is both acceptable to DOS as well as usable to generate an escape character to the printer.

Examples covered in this section assume that an IBM ProPrinter or Epson LQ-850/1050 compatible printer is connected to your computer. You will see how to create several batch files that use common printer control codes to set different printer modes of operation. If your printer does not support these control codes you can examine your printer manual to determine the appropriate codes to use in the examples presented in this section.

Compressed Printing

Assume you desire to create a batch file that can be used to print text in a compressed mode. You could create the batch file named COMPRESS.BAT, which is shown in Listing 10.2.

Listing 10.2 COMPRESS.BAT

```
C>COPY CON: COMPRESS.BAT
ECHO <ALT>15>LPT1:
TYPE %1>LPT1:
ECHO <ALT>18>LPT1:
```

Here the symbol <ALT> denotes that the user should press the ALT key while typing the numeric value following the symbol from the numeric keypad. Thus, ECHO <ALT> 15 generates the ASCII character whose decimal value is 15. When this character is routed to the printer it turns on *compressed printing,* which is also known as *condensed printing.* The second line in the COMPRESS.BAT file causes an ASCII text file, assigned to the %1 replaceable parameter, to be routed to the printer, and the third line in the file disables compressed printing.

As an alternative to using the ALT key and numeric keypad, you can enter many printer control codes directly by using the control key and an uppercase letter. As an example, consider the COMPRESS.BAT program in Listing 10.2. Because ALT+15 is a Control+O, whereas ALT+18 is a Control+R, you can enter those control characters directly. When you do so, the circumflex accent (caret, shown as ^) preceding the letter on the screen signifies that the letter is a control character. The following shows how you can directly enter control characters in the COMPRESS.BAT file to enable and disable compressed printing.

```
C>COPY CON: COMPRESS.BAT
ECHO ^O >LPT1:
TYPE %1 >LPT1:
ECHO ^R >LPT1:
^Z
```

```
        1 File(s) copied
C>
```

To illustrate the usefulness of COMPRESS.BAT, assume you create a file with EDLIN that has one or more lines of text that exceed 80 columns in length. Figure 10.1 shows the contents of the file SAMPLE.TXT, which meets the preceding criteria.

Now use the previously created COMPRESS.BAT file to print the sample text file. To do so you would enter

```
COMPRESS SAMPLE.TXT
```

Issuing that command invokes the file COMPRESS.BAT, with SAMPLE.TXT assigned to the %1 replaceable parameter. This switches the printer to compressed print mode, prints the contents of SAMPLE.TXT, and disables the compressed print mode to permit resumption of normal printing. As a result of the preceding operations the two-line file is

```
The quick Brown Fox jumped over the lazy dog who was sunbathing in the shade when the moon was bright.
```

Near-Letter-Quality Printing

If you do not have an NLQ dot-matrix printer, consider printing your text files in either emphasized or double-strike mode to obtain bolder looking text. As indicated in Table 10.1, ESC E and ESC F select and cancel emphasized printing, whereas ESC G and ESC H select and cancel double-strike printing.

Because each of the methods required to enable and disable double-strike and emphasized printing requires the use of an escape character, you can use **ALT+155** to enter the required escape code in a batch file. Then, you can follow the escape code by the uppercase character that represents the second part of the code sequence required to place the printer into a desired print mode. Thus, you could enter the sequence ALT+155+G, which when echoed to the printer turns on double-strike printing.

Listing 10.3 illustrates the creation of a batch file named DBLSTK.BAT, which places the printer into double-strike mode, prints a desired text file, and then cancels the double-strike mode, thereby returning the printer to its normal print mode. You will note that DBLSTK.BAT is very similar to the COMPRESS.BAT program previously discussed; the only differences between programs are in the printer control codes used in each program.

Listing 10.3 DBLSTK.BAT Program

```
ECHO <ALT>155G > LPT1:
TYPE %1 >LPT1:
ECHO <ALT>155H > LPT1:^Z
```

Figure 10.1
SAMPLE.TXT File

```
C>edlin sample.txt
End of input file
*1
       1:*The quick Brown Fox jumped over the lazy dog who was sunbathing in th
e shade when the moon was bright.
*
```

Likewise, you can print the SAMPLE.TXT file in double-strike mode by entering

```
DBLSTK SAMPLE.TXT
```

Rather than create a third separate batch file to perform emphasized printing, now create one program that incorporates several print modes. Then you can use one replaceable parameter to select the desired print mode and a second replaceable parameter to select the file to be printed.

Listing 10.4 is a program named PRTCHNG, which can select one of three print modes, print a file in that print mode, and then return the printer to its normal print mode.

Listing 10.4 PRTCHNG.BAT File

```
ECHO OFF
IF %1==COMPRESS GOTO :COMPRESS
IF %1==DBLSTK GOTO :DBLSTK
IF %1==EMPHS GOTO :EMPHS
ECHO WRONG PRINT MODE SELECTED
GOTO :END
:COMPRESS
ECHO <ALT>15 > LPT1:
TYPE %2 > LPT1:
ECHO <ALT>18 > LPT1:
GOTO :END
:DBLSTK
ECHO <ALT>155G > LPT1:
TYPE %2 > LPT1:
ECHO <ALT>155H > LPT1:
GOTO :END
:EMPHS
ECHO <ALT>155I > LPT1:
TYPE %2 > LPT1:
ECHO <ALT>155J > LPT1:
:END
```

The format of the command line entry to execute the PRTCHNG.BAT program is

```
PRTCHNG printmode filename
```

Here *printmode* must be COMPRESS, DBLSTK, or EMPHS, which define three named routines in the program that initiate compressed, double-strike, and emphasized printing, respectively.

After line 1 turns the echoing of commands off, the three IF statements check the value of the first replaceable parameter entered in the command line. If it equals COMPRESS, DBLSTK, or EMPHS, a branch to the appropriate location in the program occurs to perform the indicated printing operation. If no match occurs, the program displays an error message, branches to the label END, and terminates. In this program the %2 replaceable parameter specifies the file to be printed. Although

the program does not check to ensure that a valid file exists, you can add an IF EXIST batch command to test for the existence of the file. Then, if the file does not exist, the program displays an appropriate error message and branches to the label END.

You can easily modify the file listed in Listing 10.4 to add other printer modes. The use of the ALT key and the technique described for entering an escape character enable you to develop other printer control routines similar to the preceding examples.

Printer Port Swapping

Suppose you have two printers to connect to your computer—a letter-quality printer and a dot-matrix draft printer. Assume you need to use each at different times, but they are connected to two different parallel ports in the back of your PC. The following example uses a batch command to call a BASIC program that can examine available printer ports and swap the printers if you want. To accomplish the swap, create a batch file named PRINTER.BAT using the DOS COPY command.

```
A>COPY CON: PRINTER.BAT
```

The only entry in this batch file is the command to call the BASIC program.

```
BASICA PRTSWAP.BAS
```

This command calls BASICA and executes a program saved in the current directory as PRTSWAP.BAS. The BASIC program begins, in lines 100 and 110 in Listing 10.5, by defining two messages to describe which printer is currently available.

Listing 10.5 BASIC Program to Swap Printer Ports

```
100 LET LQMSG$="Current printer is LETTER-QUALITY PRINTER"
110 LET DMMSG$="Current printer is DOT-MATRIX PRINTER"
120 DEF SEG=%H40
130 LET A=PEEK(8)
140 LET B=PEEK(9)
150 LET C=PEEK(10)
160 LET D=PEEK(11)
170 CLS
180 PRINT "SWAP BETWEEN LETTER-QUALITY AND DOT-MATRIX PRINTERS"
190 PRINT"-------------------------------------------------------"
200 PRINT
210 IF A=120 AND B=2 THEN PRINT LQMSG$
220 IF A=188 AND B=3 THEN PRINT DMMSG$
230 PRINT
240 PRINT "would you like to change printers? (Y/N)";
250 INPUT X$
260 IF X$="N" OR X$="n" THEN GOTO 310
270 POKE 8,C
280 POKE 9,D
290 POKE 10,A
300 POKE 11,B
```

```
310 SYSTEM
320 END
```

Lines 120 through 160 define the base address segment for BASIC to use and PEEK commands to store the values in locations 8 through 11 of the memory address segment. These locations are where the pointers to the parallel ports controlling the two printers are stored. The PEEK command looks at the actual values in certain locations of memory in the computer, and the LET command assigns those values to the variables A, B, C, and D.

Lines 170 through 200 simply clear the screen and tell you what the program is supposed to accomplish. In programming terms, this is known as *placing a heading on the screen.*

Lines 210 and 220 check the values in the memory areas that were "peeked" at earlier and saved in the variables A, B, C, and D, and compares these values to the values relevant to your letter-quality or dot-matrix printer. Line 240 then prints a message telling you which printer is currently active. Note that your system may use different values; after you install two printers, load BASIC and use the PEEK commands as direct statements to look at the values in these memory locations. Use the values returned in A, B, C, and D in the program.

Lines 230 through 260 ask you if you would like to change printers and if your answer is (N)o, the program terminates.

If your answer is (Y)es, lines 270 through 300 swap the printers by reversing the values that were previously in the memory locations that were "peeked" at.

Line 310 ends the program and returns control to DOS and the batch commands that originally started this program.

To execute the program, simply enter **PRINTER** at the DOS prompt.

Printer Spooling

The term *spool* comes from an old procedure used on mainframes called Simultaneous Peripheral Operation On-Line. The purpose of the procedure is to prevent a slow peripheral, such as a printer or modem, from hogging all of the computer's resources until the peripheral's task is completed. Nowadays, "spooling" refers to any of several methods of transferring data from a file to a printer in such a way that you can run other programs while the printer is running.

Instead of sending output from an application program directly to the printer, character by character, at the rate the printer can accept it (50 characters per second for daisy-wheel printers, 120 to 400 cps for dot-matrix printers), you send as much output as possible to a memory buffer at high speed (many thousands of characters per second). The printer takes characters at its own speed from the buffer, leaving the computer free to perform other tasks until it needs to refill the print buffer with another block of output.

There are two main classes of print spoolers. One is an external unit that has its own microprocessor and memory. This microprocessor runs a program to get blocks of text from the host computer and feed them to one or more printers. The other type of spooler uses part of the host computer's own memory as a print buffer and software that sends characters to the printer during times when no other task of higher priority

is running (for example, while the computer is waiting for you to enter characters from the keyboard). Some printers have internal buffers that can hold 8,000, 16,000, or even 32,000 characters; such printers enable you to reduce, or even eliminate, the use of the computer's main memory for print buffers. This chapter is mainly concerned with spoolers that use part of your computer's memory for buffering.

DOS PRINT Command

In recent releases of DOS, the designers have recognized the need for a print spooler of some type and have added the PRINT command to the operating system. The PRINT command can print a file on a line printer attached to the computer. It can also create a holding area to queue up multiple files for printing and can select which device is to be used for printing if more than one printer is available on a computer. In fact, under DOS 4.0 the INSTALL program places the PRINT command in the AUTOEXEC.BAT file and specifies that the LPT1 printer port is attached to your printer by including the /D:LPT1 parameter in the command line.

Unless the /D parameter is used in the command line, when the PRINT command is first invoked it asks you to verify what the name of the device is and where the printed output is to go. The DOS default for this is PRN and this is the device selected if the ENTER or RETURN key is pressed. Other devices can include the console screen (CON), a different line printer (LPT2), or even a communication port (COM1). Once you select the desired output device, the PRINT command installs the PRINT routines in a resident part of memory and tells whether or not something is in the queue to be printed, as illustrated here.

```
C>print
Name of list device [PRN]:
Resident part of PRINT installed
PRINT queue is empty
```

To print more than one file at a time on the printer, you can simply issue the PRINT command with the desired filenames after it. The following shows the PRINT command with five different filenames after it. The files are placed in the print queue in the order specified by the PRINT command. DOS also lists all files in the queue for reference purposes.

```
C>print menu.bat 1.bat 2.bat autoexec.bat config.sys

    C:\MENU.BAT is currently being printed
    C:\1.BAT is in queue
    C:\2.BAT is in queue
    C:\AUTOEXEC.BAT is in queue
    C:\CONFIG.SYS is in queue
```

If you were using the PRINT spooler and wanted to know what files were in the queue for printing, you could simply issue the PRINT command without any parameters, as shown here.

```
C:\MENU.BAT is currently being printed
C:\1.BAT is in queue
C:\2.BAT is in queue
C:\AUTOEXEC.BAT is in queue
C:\CONFIG.SYS is in queue
```

The result is a listing of all files in the queue, in the order in which they will be printed. The PRINT command provides additional parameters you can use to add files to or delete files from the print queue.

As an extra example of the power of the PRINT command, suppose each week you use a word processor to change your budget and then print the budget with the most recent changes. You could then use the PRINT command in a batch file to automatically send a particular file to the printer immediately after it is updated. Suppose you want to again do work with your word processor while the budget is being printed. An example of a batch file to accomplish this task is shown in Listing 10.6. This batch file is called BUD.BAT.

Listing 10.6 BUD.BAT File

```
1:*cd \mm
2: mm
3: cd \
4: print <device.prn budget.prn
5: cd \mm
6: mm
7: cd \
```

Line 1 in the file changes the directory to the word processing directory, whereas line 2 executes the word processor program, which here is called `mm`—the actual name of the executable program that invokes a well-known word processor. Line 3 returns control to the root directory so the PRINT command can be used. (This is not necessary if you have previously set up a PATH command to point to where the DOS PRINT command is located.) Line 4 issues the PRINT command and specifies the file to be printed. Lines 5 and 6 change back to your word processing directory and again execute the word processor program, thereby allowing you to work while the printing is taking place. Finally, line 7 in the batch file merely returns you to the root directory when the word processing work is completed.

Although most of the entries in the batch file are self-explanatory, line 4 requires some additional elaboration. The PRINT command is there, along with the file to be printed (BUDGET.PRN), but what is the `<DEVICE.PRN` in the middle of this command? Well, remember that the PRINT command must know which device you want to print to (usually PRN) and it will always ask you the question the first time you use PRINT. You don't have to answer the question, because the < symbol tells DOS to accept its next input (the answer to the question `Name of list device`) from a file called DEVICE.PRN. This file has been placed in the root directory (that is why the batch file went back to the root directory) and looks like the following. Notice that this file has only one line, containing the value PRN:.

```
C>edlin device.prn
End of input file
*1
     1:*PRN:
*
```

The PRINT command takes the PRN: from the file DEVICE.PRN to answer the `Name of list device` question and begins printing the BUDGET.PRN file. The batch file does not stop for the user to answer any questions and continues to restart the word processor. An example of the execution of part of this batch file is

```
C>cd \
C>print <device.prn budget.prn
     C:\BUDGET.PRN is currently being printed
C>cd \mm
```

One of the key advantages of the use of the DOS PRINT command is the ability to obtain a multitasking capability. That is, you can select one or more files for printing and continue to perform other computer operations while the printing occurs. By tailoring the example presented in this section to your particular application requirement, you should be able to print either predefined files or add one or more replaceable parameters to the previously presented programs. This enables you to select different files for printing by passing the filename(s) as a replaceable parameter to the PRINT command in your batch file.

11 / OS/2

This chapter explores IBM's second operating system for personal computers—OS/2. As previously explained in Chapter 3, OS/2 is actually a set of four operating systems, two of which are text-based versions, whereas the other two versions include a graphical interface. Because the graphical interface versions of OS/2 function as a superset to the text versions, this chapter focuses on the operation and use of OS/2 by examining many features common to all versions of this operating system.

Versions of OS/2

The first version of OS/2 is IBM's OS/2 Standard Edition 1.0. Essentially, OS/2 Standard Edition 1.0 is a dual-mode operating system, providing you with the ability to execute one application at a time in its DOS mode or execute multitasking applications in each of up to 12 sessions in its multitasking mode. As you will see later in this chapter, the command interface of the OS/2 Standard Edition 1.0 is very similar to DOS; however, it does not include the graphical user interface known as the Presentation Manager. This graphical interface became available in October 1988, when IBM released OS/2 Standard Edition 1.1, which is essentially Standard Edition 1.0 of OS/2 with an enhanced user interface.

IBM's OS/2 Extended Edition 1.0—like the firm's Standard Edition 1.0—incorporated a text-based interface. The primary differences between the Extended Edition and the Standard Edition of OS/2 are

- The Extended Edition includes a database management program and a variety of built-in data communications support programs.
- Extended Edition 1.1 added the Presentation Manager graphics interface to the text interface version of Extended Edition 1.0.

Hardware Requirements

The key to multitasking and the use of memory beyond 640K bytes is operating the computer's microprocessor in its protected mode. Thus, OS/2 is restricted to 80286- and 80386-based computer systems whose microprocessors support protected mode operations. This restriction precludes the 8086-based PS/2 Model 25 and Model 30 from operating OS/2.

For PS/2 users with Intel 80286 or 80386 microprocessors, two key constraints with respect to using OS/2 are disk storage capacity and RAM memory. To install OS/2 Standard Edition on your fixed disk, you will need approximately 4M bytes of available disk space. For OS/2 Extended Edition, your disk storage requirements depend on the parts of the operating system you need and request to be installed. Under a worst case scenario, if you install the entire operating system to include its variety of communications programs, you may need 20M bytes of disk storage. RAM memory required by OS/2 Standard Edition involves a minimum of 2M bytes, whereas the Extended Edition requires a minimum of 3M bytes. Both minimum memory requirements provide the capability to efficiently use only a portion of the full multitasking capability of OS/2. As an example, if you initiated a full complement of 12 multitasking sessions, with each session using 1M byte of RAM, you might expect OS/2 to require approximately 14M bytes of memory, including 640K bytes for executing one DOS program at a time under the OS/2 DOS mode as well as room for OS/2 programs. In actuality, OS/2 uses a portion of disk memory to swap programs in and out of RAM, reducing your actual RAM memory requirements. Even so, if you do not expand RAM to 4M bytes or more to service a large number of simultaneously executing programs, *thrashing* may occur. When this situation arises, OS/2 will be spending more time moving programs and data from RAM to disk and vice versa than actually executing programs, causing performance to suffer. Due to this, you should consider expanding RAM memory to 4M bytes if you anticipate executing several large programs at one time.

If you have added memory to your PS/2, when you power-on your computer you see the error message 165 displayed on your screen, followed by a circle with the letters OK crossed out by two lines. This message indicates that the additional memory has not been recognized by your computer. At this time you should insert the Reference Diskette that came with your system, perform a system reset operation, and select "automatic configuration" from the menu that will be displayed. This will result in the additional RAM being recognized by your PS/2, permitting you to install and use OS/2.

System Installation

Each version of OS/2 includes an Installation Diskette whose use simplifies the installation process to a 10- to 20-minute task depending on which version of OS/2 you are installing and whether or not you previously installed DOS on your fixed disk. If you previously installed DOS, OS/2 does not require you to reformat your fixed disk, eliminating a few minutes from the installation process. However, note that once OS/2 is installed on your fixed disk, simply removing those files will not enable you to self-boot using DOS. You either have to purchase a third-party utility program or back up your files, reformat your disk, and then reinstall your files.

The initial installation of OS/2 Standard Edition takes approximately 10 minutes if you previously installed DOS. You simply place the diskette labeled Install in drive A and press the **Ctrl+Alt+Del** multikey combination to perform a system restart (or power-on your PC if it was previously turned off). Then you can follow a series of menus that will guide you through the installation process.

After the display of a copyright notice, you are informed that OS/2 renames a previously created CONFIG.SYS file as CONFIG.BAK, whereas a previously created

AUTOEXEC.BAT file is renamed AUTOEXEC.BAK. The extension .BAK will be used unless you enter different filenames for those files created under DOS. OS/2 renames the CONFIG.SYS and AUTOEXEC.BAT files as it creates two new files with those names to govern the initial operation of this operating system.

During the installation process you are prompted when to insert and remove the different diskettes packaged with your version of OS/2. As the contents of each diskette are copied to drive C, the message `Copying from IBM Operating System/2 Diskette n` is displayed. As data is being copied from diskette to your fixed disk, it is also being organized under directories that either previously existed, such as the root directory, or that are new directories created during OS/2 installation.

The directories created by the OS/2 Standard Edition include C:\OS2, C:\OS2\INSTALL, C:\OS2\INTRO, and C:\SPOOL, whereas the directory C:\ is created only if your fixed disk required formatting. Otherwise, OS/2 uses C:\ without having to create a root directory. OS/2 system files are placed into the root directory (C:\), and system utilities and installable device drivers are stored in the OS2 subdirectory (C:\OS2). The subdirectory \SPOOL is the default location where a printer spooler program included in OS/2 (SPOOL.EXE) temporarily stores data prior to printing it. The spooler operates by capturing all printer output as files to disk, then transferring the contents of the files to your printer as a background task under the multitasking capability of OS/2.

Similar to the installation of DOS 4.0, OS/2 displays several screens you use to select the country and keyboard setting and printer and printer port assignment. Other screens displayed during the installation process enable you to select a mouse and serial device. Selection of the latter is required, because OS/2 must keep track of the use of such serial devices as plotters and/or printers to enable them to be correctly used by two or more programs that require access to the same device.

During the installation process you can view and modify the Mode Configuration for OS/2 and DOS. Figure 11.1 illustrates the default OS/2 Mode Configuration screen, while Figure 11.2 illustrates the DOS Mode Configuration screen displayed under OS/2.

In examining the OS/2 Mode Configuration, you probably recognize the BUFFERS entry. BUFFERS is an OS/2 configuration command used in an OS/2 CONFIG.SYS file similar to the way it is used in a DOS configuration file. That is, the command defines the number of 512-byte blocks of storage known as *disk buffers* that OS/2 will

Figure 11.1
OS/2 Mode Configuration Screen

```
                    OS/2 Mode Configuration

BUFFERS. . . . :  30                (1 - 100)
DISKCACHE. . . :  64                (64 - 7200)
MAXWAIT. . . . :  3                 (1 - 255)
MEMMAN MOVE. . :  MOVE              (MOVE/NOMOVE)
MEMMAN SWAP. . :  SWAP              (SWAP/NOSWAP)
PRIORITY . . . :  DYNAMIC           (DYNAMIC/ABSOLUTE)
PROTECTONLY. . :  NO                (YES/NO)
SWAPPATH . . . :  C:\
THREADS. . . . :  64                (32 - 255)
TRACE. . . . . :  OFF               (ON/OFF)
TRACEBUF . . . :  4                 (1 - 63)
```

Figure 11.2
OS/2 DOS Mode Configuration Screen

```
                    DOS Mode Configuration

BREAK. . . . . :   OFF    (ON/OFF)

OPEN FCBS. . . . . :   16     (1 - 255)

PROTECTED FCBS . . :   8      (0 - 255)     Must be less than or
                                            equal to OPEN FCBS
RMSIZE . . . . . :   640    (256 - 640)
```

use to read and write blocks of data that do not occupy an entire sector. The other entries in Figure 11.1 represent the default settings of configuration commands unique to OS/2, which the system installs in its CONFIG.SYS file. As an example, the DISKCACHE command allocates a specified number of 1024-byte blocks between 64 and 7200 that are taken out of storage above 1M byte. This area is used to increase the speed of I/O operations, because it acts as an intermediary between storing and retrieving data to/from disk and application programs. As an example, a program can write data to the cache at electrical speed and continue processing while a background task writes data from memory to the disk at electromechanical speed. Later in this chapter OS/2 configuration commands are covered.

Once your version of OS/2 has been successfully installed, you are prompted to remove any diskette in drive A and press **Ctrl+Alt+Del** to start OS/2. After you do so, OS/2's Program Selector screen, which is the operating system's main menu, is displayed. This screen has a highlighted bar across the top labeled `Updated` at the left and `F1=Help` at the right and can be considered as a rudimentary graphical user interface, displaying two boxes as illustrated in Figure 11.3.

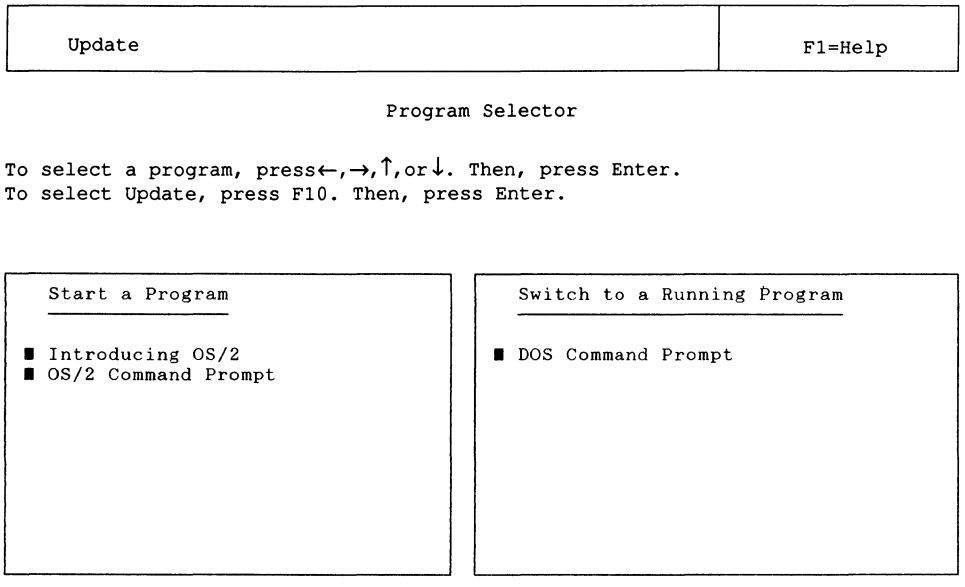
Installation Tips

If you installed DOS prior to installing OS/2, you will have multiple copies of numerous OS/2 commands that are "dual mode" and that operate under OS/2 mode and DOS mode. Due to this, if you previously placed DOS commands under the subdirectory \DOS, you may wish to consider removing all files in that subdirectory as well as that directory. This is because OS/2's DOS mode can use those commands that were installed in the subdirectories \OS2 and \OS2\INSTALL to perform the same DOS functions.

The Program Selector

The Program Selector can be considered as OS/2's main menu. Through the use of the `Start a Program` and `Switch to a Running Program` box entries, you can access the OS/2 or DOS command prompt mode of operation. You can also start programs, switch between running programs, add titles of installed programs, and define their location for automatic execution when the program name is selected as well as add or change program information on the Program Selector.

Figure 11.3
Program Selector Main Menu



You can move the highlighting bar with an arrow key to an entry in either box and press the **Enter** key to effect your selection. When you select `OS/2 Command Prompt` you can directly enter OS/2 commands. To ensure that you remember you are in the OS/2 command prompt mode of operation, the OS/2 system prompt is displayed in brackets. Thus, if C is your default drive and you are at the root directory, the OS/2 prompt would be displayed as

```
[C:\]
```

To return to the Program Selector main menu, you can press **Ctrl+Esc**. If you select the DOS Command Prompt entry, you can enter DOS commands. This action displays what many persons refer to as the "DOS compatibility box," from which you can execute DOS programs under OS/2. This is why `DOS Command Prompt` is a default entry in the Switch box.

When the DOS Command Prompt selection is made, the system prompt is the familiar DOS prompt. That is, if drive C is the default drive and you are located at the root directory, your system prompt will be

```
C:\>
```

Similar to exiting the OS/2 Command Prompt, you can press **Ctrl+Esc** to access the Program Selector.

The first option listed in the Start a Program box—`Introducing OS/2`—is a brief tutorial. It gives an overview of how multitasking under OS/2 runs as many as 12 OS/2 programs at the same time, how to use the Program Selector, and how to use the OS/2 Help facility.

Adding OS/2 Programs

You can add OS/2 programs to the Start a Program box through a menu system initiated when you press **F10** and **Enter**, or Mouse Button 1 when the Program Selector screen is displayed. Performing either operation displays the update menu illustrated in Figure 11.4.

Selecting option 1 from the Update menu displays the Add a Program Title screen. This screen is illustrated in Figure 11.5, with entries included for the program title and program pathname.

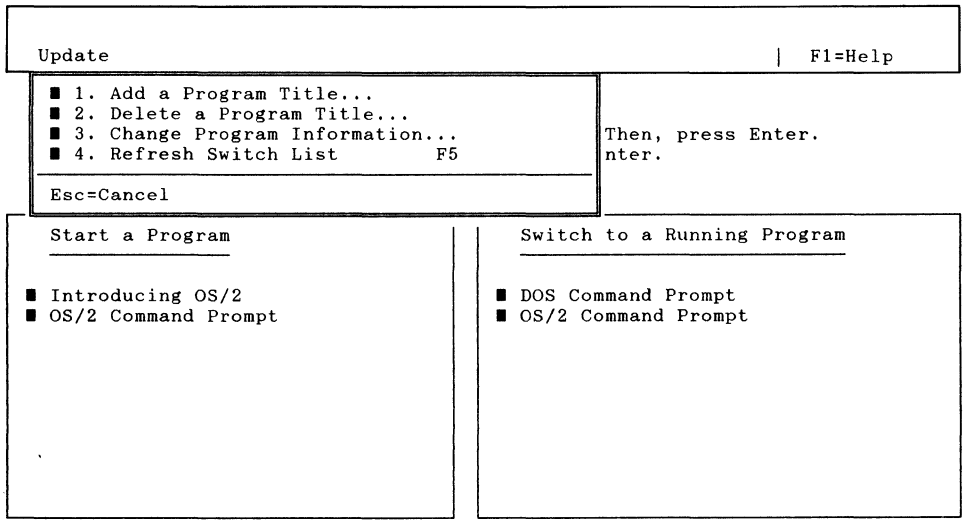


Figure 11.4
Program Selector Update Menu

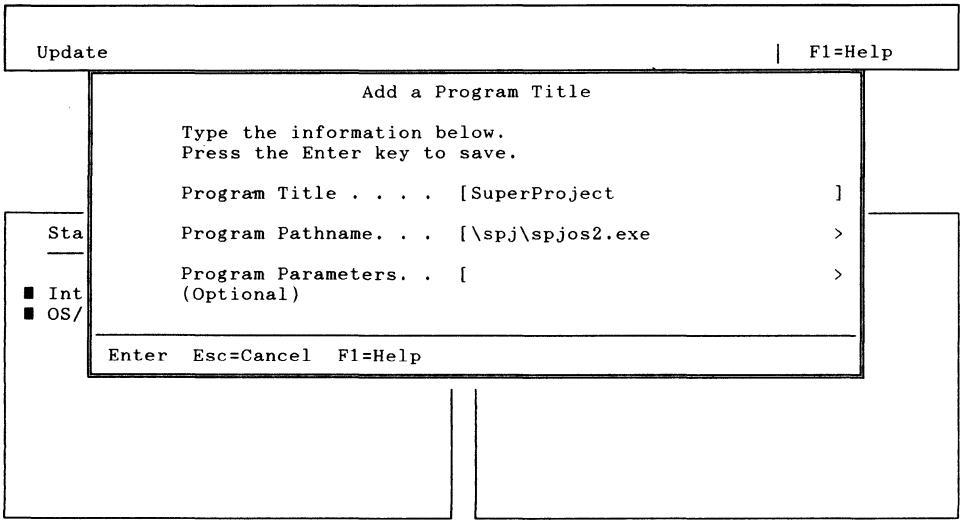


Figure 11.5
Add a Program Title Screen

The program title is the name you want listed on the Start a Program box and can be up to 30 characters in length. The `Program Pathname` is the location of the program on a disk or diskette, including the drive designator, directory, and executable filename. In the example illustrated in Figure 11.5, the file SPJOS2.EXE in the subdirectory SPJ will be executed when the program SuperProject is selected from the Start box. (SuperProject is a sophisticated project management program from Computer Associates International, Inc., which includes PERT and Gantt charting capabilities.)

The `Program Parameters` entry defines an action or set of actions you want to occur each time you select the program. As an example, you might select a parameter associated with a word-processing program that causes a legal spelling checker to be used with the program. If you prefer to have the system prompt you for the parameters to use, you can enter the question mark (?) character into this field, so you can change the parameters each time you select the program. Once you complete entering data into the fields of the Add a Program Title screen, pressing **Enter** adds the title to the Start a Program box as well as returns you to the Program Selector main menu that displays that box.

Although the first three options in Figure 11.4 are essentially self-explanatory, option 4, `Refresh Switch List`, deserves an explanation. The Start a Program box lists programs that have started. If you wish to update the list to display only currently running programs, you can select option 4 or press **F5**. This switches between running programs, because the Start box will only display those programs in execution.

Once a program title is added, the Program Selector Start box reflects the new entry. This is illustrated in Figure 11.6, which shows the result of adding the SuperProject program. Once this program is placed into the Start box, you invoke it by moving the highlight bar over the program name and pressing **Enter** after the appropriate program files are placed under the appropriate subdirectory.

Directory and File Operations

OS/2 and DOS programs and data files use a common file-system format, which enables media written by one operating system to be read by the other. Similar to DOS, OS/2 can organize files into tree-structured directories. In fact, both file-naming conventions and drive designators are the same in OS/2 as under DOS. OS/2 also locks files that are used by OS/2 mode or DOS mode programs to prevent other programs from altering

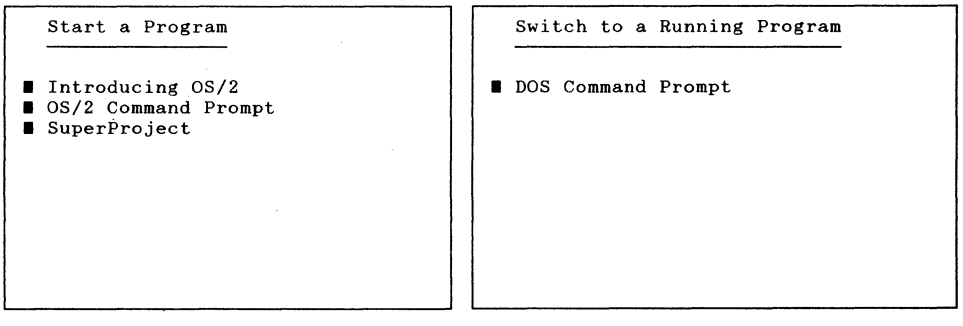


Figure 11.6
Revised Program Selector Boxes

their contents. Due to this, directory and file reference commands are the same for both OS/2 and DOS.

Table 11.1 lists 15 directory and file reference commands that function similarly under OS/2 and DOS, including operating DOS under OS/2. The key difference between most OS/2 and DOS file reference commands is the ability of six OS/2 commands to accept multiple arguments. OS/2 versions of the DIR, ERASE, MKDIR, RMDIR, TYPE, and VOL commands accept multiple arguments. As an example,

MKDIR C:\OS2\UTILITY C:\DOS\APP

creates the subdirectory UTILITY under the OS2 directory and the subdirectory APP under the DOS directory on drive C.

Similarly,

DIR STAT.* *.DAT

produces a directory listing of all files whose name is STAT and all files whose extension is .DAT. Refer to Chapters 5 and 6 for information concerning the basic syntax and operation of each of the commands listed in Table 11.1. To illustrate the use of a few of these commands under OS/2, this sample session has you select the OS/2 Command Prompt from the Start box and create a directory named SPJ. Then, you use the COPY command to initiate the transfer of files from the distribution diskettes included with the SuperProject program to the directory just created.

The following illustrates what interchange occurs on the initial OS/2 Command Prompt screen. The top line is displayed within a highlighted bar that remains on the

Table 11.1 Common OS/2 and DOS Directory and File Reference Commands

Command	Function Performed
BACKUP	Creates backup copies of one or more files.
CD or CHDIR	Changes the current directory.
COMP	Compares the contents of two files.
COPY	Copies the contents of files from one disk or directory to another disk or directory.
DIR	Lists files and subdirectories in a directory.
DISKCOMP	Compares the contents of two diskettes.
DISKCOPY	Compares the contents of one disk to another.
ERASE	Removes a file from disk.
MD or MKDIR	Creates a directory.
RD or RMDIR	Removes a directory or subdirectory.
RENAME	Changes a filename and/or its extension.
RESTORE	Restores one or more files previously backed up.
TREE	Displays the directory structure and optionally lists the files in each subdirectory.
TYPE	Displays the contents of a file.
VOL	Displays the disk volume label if one exists.

screen as long as you use the OS/2 Command Prompt mode or until you turn off the OS/2 help facility. The first entry in the bar—OS/2—reminds you that you are working in that operating system. Next, the message prompt `Ctrl+Esc = Program Selector` informs you how to return to the main menu, and the last prompt tells you how to invoke HELP.

```
    OS/2          Ctrl+Esc = Program Selector                Type HELP = help
[C:\]PATH C:\;C:\OS2;C:\OS2\INSTALL;

[C:\]DPATH C:\;C:\OS2;C:\OS2\INSTALL;

[C:\]CALL HELP ON
```

The `PATH` display informs you that the root directory, the OS2 subdirectory, and the subdirectory INSTALL under the OS2 directory search for commands and programs not found in the current directory. The `DPATH` command, which is only applicable to OS/2, creates a search path for data files outside the current directory to the same locations as the PATH command. Finally, the `CALL HELP ON` prompt informs you that the OS/2 help line is enabled. This causes the top line of the display to tell you how to return to the Program Selector menu, switch to the next session when two or more sessions are active, exit the current session, and obtain additional help on error and warning messages. To remove the help line, you can enter the command HELP OFF. Similarly, if you decide you need the help line, you can turn it back on by entering HELP ON.

When you use the OS/2 Command Prompt mode, you can enter commands similar to the manner in which DOS commands are entered. In fact, the responses from executing OS/2 commands, in many instances, appear exactly the same as if a DOS command was executed, due to the commonality of commands used by both operating systems. This commonality is illustrated in the help example shown here of the creation of a directory under OS/2, changing the current directory to the newly created directory, and copying the contents of two diskettes into the new subdirectory. Note that other than the OS/2 prompt—indicated as such by being enclosed in brackets—the functions performed and resulting display are the same for both operating systems.

```
[C:\]md\spj

[C:\]cd spj

[C:\SPJ]copy A:*.* c:
A:SPJOS2.EXE
A:READ.ME
A:SPJGRAPH.DAT
A:SPJGRAPH.EXE
A:SPJPRINT.DAT
              5 file(s) copied.

[C:\SPJ]copy A:*.* c:
```

```
A:TUTORIAL.XQT
A:SIDEWAYS.COM
A:SWSETUP.COM
A:PRCONFIG.EXE
A:DEMO.XQT
A:SPJ.HLP
A:LEVELING.PJ
A:LINKS.PJ
A:ELM_HIGH.PJ
A:MOVE.PJ
              10 file(s) copied.

[C:\SPJ]
```

Now that the SuperProject files are located in the subdirectory \SPJ, you can initiate the program from the Start box in the Program Selector menu. This executes the file SPJOS2.EXE, and the SuperProject program is run.

Suppose you just created a large PERT chart you wish to print prior to performing additional work. At the same time, assume you want to create a Gantt chart for a different project. You can accomplish both tasks and more under OS/2 due to its multitasking capability.

Without terminating SuperProject, you can press **Ctrl+Esc** to return to the Program Selector main menu. The Start and Switch boxes now appear as illustrated in Figure 11.7. Note that `SuperProject` is displayed in the Switch box as it is currently running. At this time, you can select `SuperProject` from the Start box and initiate a second SuperProject program while the first program remains active. In fact, you can easily switch between the two SuperProject programs and initiate up to 10 additional OS/2 programs, as well as a DOS program. Because DOS and OS/2 commands are programs, you can even use the FORMAT command to initialize a diskette in addition to running multiple versions of the SuperProject program and performing other activities.

To terminate an OS/2 SuperProject session, select its Quit option, which returns you to the Program Selector main menu. If you are in the OS/2 Command Prompt mode, you can type EXIT to end the OS/2 session. Table 11.2 summarizes the use of OS/2 HELP prompts displayed on various screens as you use OS/2. These prompts can also be displayed if you type **HELP** from the OS/2 command prompt mode.

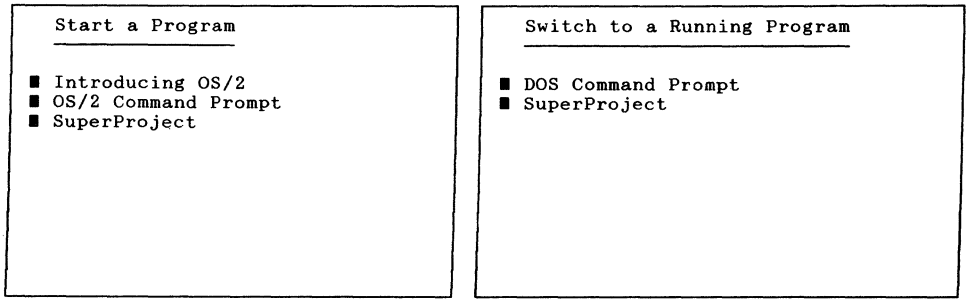


Figure 11.7
Revised Start and Switch Boxes

Table 11.2
OS/2 HELP Prompts

```
[C:\]help
Alt+Esc to switch to the next session.
Ctrl+Esc to switch to the program selector.
Type HELP ON for help text.
Type HELP OFF for no help text.
Type HELP message-number for message help.
Type EXIT to end this OS/2 session.
[C:\]
```

Table 11.3
Command Line Operator Support

CMD.EXE	COMMAND.COM	Function
^		Lexical Escape Character
<	<	I/O Redirection (Input)
>	>	I/O Redirection (Output)
>>	>>	I/O Redirection (Output Append)
¦	¦	Pipe Operator
&&		AND Operator
¦¦		OR Operator
&		Command Separator
( )		Command Grouper

OS/2 Command Prompt Operations

OS/2 comes with two command processors—COMMAND.COM and CMD.EXE. COMMAND.COM is the real-mode command processor similar to the familiar DOS command processor. As in DOS, COMMAND.COM is used to start programs in the DOS Command Prompt mode.

CMD.EXE is the OS/2 command processor that has extended the functionality of the command language by including additional command line operators, as well as support for COMMAND.COM operators. In addition, CMD.EXE provides enhanced batch file support.

Table 11.3 lists the command line operators supported by CMD.EXE and COMMAND.COM. In interpreting the data presented in this table, those operators supported by CMD.EXE are command line operators you can use in the OS/2 Command Prompt mode. The operators supported by COMMAND.COM can be used under DOS or from the DOS Command Prompt mode under OS/2.

The pipe and I/O redirection operators function under OS/2 in the same manner as under DOS and the DOS mode under OS/2. In fact, OS/2 includes the same three filters supported by DOS—SORT, FIND, and MORE. You can refer to Chapter 9 for information concerning the general use of these operators and filter commands.

Escape Operator

The escape operator (^) enables you to employ significant characters as literal text. When used during command line input, the ^ symbol causes the character following that symbol to be treated as text.

As an example of the ^ character, assume you are developing a batch file to explain the use of I/O redirection symbols. You could use the ECHO command as follows:

```
ECHO THE ^> SYMBOL IS USED FOR OUTPUT REDIRECTION
```

When you use the ^ symbol, the message THE > SYMBOL IS USED FOR OUTPUT REDIRECTION would be displayed.

AND Operator

The AND operation is defined by the use of two ampersand symbols (&&). When you use the AND operator, you can make command processing dependent on the success or failure of other commands on the command line.

The AND operator is used between two command operators:

command operator && *command operator*

If the command on the left side of the AND operator is successful, the AND operator will cause the command to its right to be performed. If the command on the left side of the AND operator fails, the command to the right of the operator is not performed. The following command line entry illustrates the use of the AND operator.

```
DIR CONFIG.BAK && TYPE CONFIG.BAK
```

In this example, the contents of the file CONFIG.BAK are displayed only if it exists in the current directory.

OR Operator

The OR operator is defined by the use of two broken vertical bars (¦¦) that result from pressing the piping symbol key (Shift+\) twice. If code page switching is in effect on your computer, the OR command operator appears as two unbroken vertical bars, even though the operator is created by pressing the piping symbol key twice.

The reverse of the AND operator, the OR operator, allows a command following it to execute only if the command preceding it failed. If the command to the left of the OR operator is successful, the command following the OR operator is not performed.

The following example illustrates the use of the OR operator.

```
COPY C:*.* A: ¦¦ DIR A:
```

Here the COPY command duplicates all files in the current directory of the fixed disk to the diskette in drive A. If the copying operation fails, the OR operator initiates the DIR command to obtain a directory listing of the contents of the diskette in drive A.

Command Separator

The command separator symbol is a single ampersand (&) character. When used in an OS/2 command line, the command separator causes the commands on both sides of the symbol to be executed in sequence from left to right, with the output from both commands occurring after the second command is executed.

The following example illustrates the use of the command separator.

```
TYPE A:PAY.DAT & C:JAN.DAT
```

This command line displays the contents of the file PAY.DAT on drive A and JAN.DAT on drive C.

I/O Redirection and Filters with Multiple Arguments

You can use I/O redirection, piping, and/or filter commands in OS/2 command lines that contain multiple arguments. Table 11.4 lists three examples of the use of I/O redirection and filter commands in OS/2 command lines that contain multiple arguments as well as the operational result of each command line entry.

Command Grouping

Using parentheses you can group internal commands together to permit them to override the default precedence and be executed in a desired sequence. To illustrate the use of command grouping, assume a diskette in drive A contains three files named FILE1, FILE2, and FILE3. Further assume that FILE1 contains the character *A*, whereas FILE2 and FILE3 contain the characters *B* and *C*, respectively. Now examine the execution of the following OS/2 command line entry:

```
DIR A:*.* >> FILE1 & (FILE1 >> FILE2 & FILE2 >> FILE3)
```

Commands are executed from left to right in a command line with parentheses determining the order of execution. In the preceding example, `DIR A:*.* >> FILE1` causes the three entries in the directory (`FILE1`, `FILE2`, and `FILE3`) to be appended to the contents of FILE1. Thus, FILE1 contains A, FILE1, FILE2, FILE3. Now examine the operation of the two commands contained in the command grouping. The `FILE1 >> FILE2` command appends the contents of FILE1 to the contents of FILE2. Thus,

Table 11.4
Using I/O Redirection and Filters with Multiple Arguments

Command Line Entry	Operational Result
TYPE A:PAY.DAT & C:JAN.DAT>LPT1	Lists the contents of the file PAY.DAT on drive A and JAN.DAT on drive C on the printer.
DIR XYZ && TYPE XYZ ¦ SORT>LPT1	If the file XYZ exists its contents are sorted and listed on the printer.
ERASE XYZ ¦¦ DIR>LPT1	If the file XYZ does not exist the current directory is listed on the printer.

the contents of FILE2 become B, A. Finally, the command `FILE2 >> FILE3` appends the contents of FILE2 to the contents of FILE3. This causes the contents of FILE3 to become C, B, A.

Now rewrite the previous command line without parentheses as follows:

```
DIR A:*.* >> FILE1 & FILE1 >> FILE2 & FILE2 >> FILE3
```

The first command, `DIR A:*.* >> FILE1`, produces the same result as in the previous command line, resulting in the contents of FILE1 becoming A, FILE1, FILE2, FILE3. Next, the command `FILE1 >> FILE2` results in the contents of FILE2 becoming B, A, FILE1, FILE2, FILE3. Finally, the `FILE2 >> FILE3` command causes the contents of FILE3 to become C, B, A, FILE1, FILE2, FILE3. Thus, significant differences in the execution of multiple commands can occur based on the presence or absence of command groupings.

Command Comparison

Because OS/2 is a dual-mode operating system, its commands can be placed into three categories. Some commands only operate in its OS/2 mode, other commands only operate in its DOS mode, while a majority of its commands operate in both OS/2 and DOS modes. Even in the latter category, there are a few commands that have increased functionality under OS/2. Most commands in this category as previously explained accept multiple arguments.

In Table 11.5, a list of OS/2 Standard Edition commands is grouped according to the operating system(s) they can be used with. Note that only internal and external commands are listed in Table 11.5, with batch commands and configuration commands discussed later in this chapter.

New OS/2 Commands

The introduction of OS/2 Standard Edition resulted in 11 commands that only operate in the OS/2 mode. Two of these commands—FDISK and KEYB—that once operated under DOS are now restricted to operating under OS/2. FDISK creates, displays, or modifies a fixed disk partition, and KEYB selects a special keyboard layout to replace the default U.S. keyboard layout. Both of these commands were moved to operate under OS/2 because this operating system is used to initialize your hardware.

Internal Commands

As indicated in Table 11.5, three internal commands were introduced with OS/2 and only operate in OS/2 mode—DETACH, DPATH, and START.

DETACH Command

DETACH executes a program in the background if it does not require keyboard input or screen output. Because OS/2 supports the use of separate segments of video memory for up to 16 applications, detaching a program that does not use standard I/O frees a

Table 11.5
OS/2 Command Comparison

Type	OS/2 and DOS dual mode		OS/2 mode only	DOS mode only
Internal Commands	CHCP	PATH	DETACH[1]	BREAK
	CHDIR	PROMPT	DPATH[1]	
	CLS	RENAME	START[1]	
	COPY	RMDIR[2]		
	DATE	SYS		
	DIR[2]	TIME		
	ERASE[2]	TYPE[2]		
	EXIT	VER		
	LABEL	VERIFY		
	MKDIR[2]	VOL[2]		
External Commands	ATTRIB	MORE	ANSI[1]	APPEND
	BACKUP	PATCH	CMD[1]	ASSIGN
	CHKDSK	PRINT	CREATEDD[1]	COMMAND
	COMP	RECOVER	FDISK	GRAFTABL
	DISKCOMP	REPLACE	KEYB	JOIN
	DISKCOPY	RESTORE	SPOOL[1]	SETCOM40[1]
	FIND	SET	TRACE[1]	SUBST
	FORMAT	SORT	TRACEFMT[1]	
	HELP	TREE		
	MODE	XCOPY		

1. Commands introduced with OS/2 Standard Edition.
2. Commands that accept multiple arguments.

segment of video memory for use by other applications. The format of the DETACH command is

```
DETACH command
```

where *command* is an external OS/2 command or program. As an example of the use of DETACH, consider the following use of that command:

```
DETACH CHKDSK > CHKDSK.OUT
```

The preceding command line causes CHKDSK to execute in the background, with its output redirected to the file CHKDSK.OUT. Because the OS/2 prompt returns immediately after you enter the command line, you can proceed to perform another operation while CHKDSK is executing. Later, you can use the TYPE command to examine the result of the CHKDSK command.

DPATH Command

The DPATH command sets a search path for data files outside of the current directory. The format of this command is

```
DPATH [{[d:] path;[path; ...]}]
      [{[;]                  }]
```

When entered without any parameters, DPATH displays the paths to data files currently in effect. If you enter the command followed by a semicolon, DPATH deletes any previously created paths to data files.

The default DPATH setting can be easily appended by following the command with the command name enclosed by percent (%) signs. As an example,

```
DPATH %DPATH% C:\UTILITY;C:\123;
```

appends the paths of the utility and 123 directories on drive C to the DPATH command currently in effect.

START Command

The START command can be entered at the OS/2 command prompt or included in a special batch file named STARTUP.CMD to start OS/2 programs automatically. STARTUP.CMD is OS/2's equivalent of DOS's AUTOEXEC.BAT file, with batch files in OS/2 having the extension .CMD in place of the .BAT extension used under DOS. The format of the START command is

```
START [string][/C] Command [arguments]
```

The string is the title you want assigned to your program. Once you start it, the string will be displayed on the Program Selector menu. The /C parameter causes the session to end when the command is complete. The command can be an OS/2 internal or external command, a .CMD batch file, or any OS/2 program you want to pass to the command processor you are starting.

The following example illustrates the use of the START statement in a STARTUP.CMD file.

```
CD \SPJ
START SPJOS2 /C
```

In this example, the directory SPJ is made the current directory. Then the START command starts the SPJOS2 program, causing the session to end when the program is completed. Because a string is not specified, the filename SPJOS2 is displayed on the Program Selector menu.

When you use the START command in the OS/2 Command Prompt mode with I/O redirection, you should use quotation marks to delimit your I/O redirection. Otherwise, CMD.EXE will interpret your command line entry to redirect the output of the START command. As an example of this, consider the following START command.

```
START "List file to printer" TYPE
C:\WP.DAT>LPT1
```

CMD.EXE interprets this command line entry as a request to redirect the output of the START command to the printer. Therefore, to redirect I/O for a specified command you should enclose the command string in quotes:

```
START "List file to printer" "TYPE
C:\WP.DAT>LPT1"
```

External Commands

Although there are nine external commands that are restricted to use in the OS/2 mode, only seven of those commands are new. As previously explained, FDISK and KEYB previously operated under DOS but were revised to operate under OS/2 as they perform initialization functions related to your hardware at system setup.

ANSI Command

The ANSI command enables or disables support for redefining keys, manipulating the cursor, and changing display color attributes in OS/2 mode. The format of this command is

```
[d:][path] ANSI [{ON}]
                [{OFF}]
```

The effect of this command is the same as that of the DEVICE=ANSI.SYS statement in a DOS CONFIG.SYS file when you set ANSI ON. In addition, you can display the state of ANSI support by entering the command without a parameter.

CMD Command

The CMD command can be used to start another command processor in OS/2 mode. The format of this command is

```
[d:][path] CMD [{[/K string]}]
               [{[/C string]}]
```

The /K string option passes the command enclosed as a string to CMD.EXE without returning to the previous command processor after the command is completed. The command represented as a string uses either blanks or quotation characters as delimiters. If the /C string parameter is used, a return to the previous command processor occurs after the command is completed. If you enter the command without parameters, another command processor is started. This command processor uses the environment that the current OS/2 command processor uses, including the default system prompt and the PATH and DPATH settings.

The following example illustrates the use of the CMD command.

```
CMD /C TYPE X.DAT>LPT1
```

The preceding command line loads another command processor. The second command processor directs the contents of the file X.DAT to the printer and returns you to the previous command processor when the command is completed.

CREATEDD (Create Dump Diskette) Command

The CREATEDD command dumps the contents of your computer's memory to diskette. The resulting dump diskette is designed to be used for IBM service personnel to assist in diagnosing system problems. The format of this command is

```
[d:][path] CREATEDD d:
```

The second drive designator indicates the target drive on which the contents of memory are to be recorded.

When you initiate the command CREATEDD, the diskette in the target drive is initialized so it can be used by OS/2's dump facility. Thereafter, you can insert the dump diskette created with the CREATEDD command in the target drive, press **Ctrl+Alt**, and press the **NumLock** key twice to start the memory dump.

SPOOL (Print Queue) Command

The SPOOL command intercepts print data from application programs, PrtSc, and I/O redirection to the printer prior to printing. During the interception process, print jobs are temporarily stored on disk, with the directory \SPOOL used as the default directory for queuing print jobs. As each print job is completed, the SPOOL program automatically generates the required number of line feeds to position the start of printing for the next print job at the top of a new page.

When OS/2 is installed, a RUN statement is inserted in your CONFIG.SYS file. That statement initiates the SPOOL command file that was placed in the OS2 subdirectory, using the printing device you indicated as available when you installed OS/2.

The format of the SPOOL command is

```
[d:][path] SPOOL [d:][directory] [ { [/D: device] } ]
                                 [ { [/O: device] } ]
```

The drive indicator and path preceding the command keyword designate the location of the SPOOL.EXE file. The drive and directory following the command keyword indicate where temporary spool files will be stored. The default is the directory \SPOOL; however, you can change the directory on a temporary basis by entering the SPOOL command at the OS/2 Command Prompt mode. To change the SPOOL directory on a permanent basis or until another SPOOL command is issued, you should change the RUN statement in your CONFIG.SYS file.

The /D: device option specifies the input print device. This is the device the application program thinks it is printing to, such as LPT1, LPT2, LPT3, or PRN. If not specified, the default is LPT1, which is the print device used by PrtSc. The /O: device option specifies the actual device on which output from an application program is printed. If not specified, the /D: device will be used as its default. In addition to specifying LPT1, LPT2, LPT3, and PRN, you can also specify COM1 through COM3 as output devices when a serial printer is attached to your computer's serial port.

The following example illustrates the use of the SPOOL command to temporarily store print jobs under the directory PRINT, using LPT1 as input and output devices.

```
SPOOL C:\PRINT /D:LPT1 /O:LPT1
```

TRACE (System Events) Command

Similar to the CREATEDD command, the TRACE command is intended as an aid for IBM service personnel. This command selectively enables or disables the tracing of

system events, such as opening a file and writing data to a file. The format of this command is

```
[d:][path] TRACE {OFF} [event code]
                 {ON }
```

The TRACE command can be placed in your CONFIG.SYS file. If not included in CONFIG.SYS, a default of TRACE OFF is assumed.

The event code is a number between 0 and 255 that indicates the major event to be traced. Numbers for major events must be requested from your IBM service representative. When a TRACE ON command occurs, the results are placed into a special buffer whose contents are displayed by the use of the TRACEFMT command.

TRACEFMT (Trace Formatter) Command

The TRACEFMT command displays the contents of the trace buffer in reverse time stamp order. Like CREATEDD and TRACE, TRACEFMT is intended for use with assistance provided by IBM service personnel. The TRACEFMT commands format is

```
[d:][path] TRACEFMT
```

The output generated by the command is designed for analysis by IBM service personnel.

New Dual Mode Commands

With the introduction of OS/2 Standard Edition, HELP and PATCH commands were added. Both of these external commands operate under OS/2 and DOS modes.

HELP Command

The HELP command displays information concerning a specific warning or error message and enables or disables the display of a help line as part of the command prompt. The format of this command is

```
                    ┌          ┐
[d:][path] HELP     │ OFF      │
                    │ ON       │
                    │ Message ID │
                    └          ┘
```

If you enter HELP without any parameters, a HELP screen will be displayed, as previously illustrated in Table 11.2. HELP ON turns on the display of the help line, whereas HELP OFF disables the display of the help line. By entering a message number, you can display specific information related to a previously displayed warning or error message. This is illustrated in Figure 11.8, in which the user attempted to execute BASICA—designed to operate under DOS—under OS/2.

PATCH Command

The PATCH command recognizes that programming is performed by humans who on occasion make errors. This command is designed to enable you to apply IBM-supplied patches to fix previously obtained programs. The format of this command is

```
[d:][path] PATCH [d:][path] filename.ext [/A]
```

Figure 11.8
HELP Facility Example

```
[C:\OS2]basica
SYS0191: C:\OS2\BASICA.COM cannot be run in IBM Operating System/2 mode.

[C:\OS2]help 191

SYS0191: *** cannot be run in IBM Operating System/2 mode.

EXPLANATION: The specified file is either a DOS
mode program or an improper program.
ACTION: If the specified program is a DOS mode program,
switch to DOS mode and retry the command.
Otherwise reinstall the application and retry the command.
If the error occurs again, contact the supplier of the
application.


[C:\OS2]
```

The drive designator and path preceding the command keyword indicate the location of the PATCH command file. The drive designator and path following the command keyword indicate the location of the specified file containing patching information. The /A option initiates the command operating in its automatic mode. In this mode, the specified filename contains instructions for patching one or more files. If you do not specify the /A option the PATCH command will prompt you through the required tasks to perform patching.

New DOS Mode Command

OS/2 included one additional DOS mode command not included in standalone DOS. This command, SETCOM40, is designed to prevent conflicts between OS/2 and DOS programs attempting to use serial ports on your computer.

SETCOM40 (Set COM Port Address) Command

The SETCOM40 command is used to enable the use of a serial (COM) port for a DOS application. The format of this command is

$$[d:][path]\ \texttt{SETCOM40 COM}n\ \left\{\begin{matrix}\texttt{ON}\\ \texttt{OFF}\end{matrix}\right\}$$

where *n* is the communications port number, 1 through 3. If you want to use a serial port under OS/2, you should not use this command. Instead, you must specify the device driver COM02.SYS in your CONFIG.SYS file. The reason for this is that OS/2 cannot manage programs using serial devices in DOS mode, requiring you to manage those programs.

The following example illustrates the use of SETCOM40 to permit output to be routed to COM1 in DOS mode.

```
SETCOM40 COM1 ON
```

Once you issue a SETCOM40 command to enable a COM port, you should not switch from DOS to OS/2 mode without first ending the DOS application that is using the COM port. Otherwise, it is possible that the application using the COM port will fail.

Standard Edition Configuration File

One of the best ways to explore the capability and operation of OS/2 is by examining its CONFIG.SYS file. Listing 11.1 contains a listing of the OS/2 Standard Edition 1.0 CONFIG.SYS file that is created if you accept the default mode configuration settings previously illustrated in Figure 11.1. If you examine the first line in Figure 11.1, you will note that the TYPE command was used to display the contents of the CONFIG.SYS file.

Listing 11.1 OS/2 Standard Edition Configuration File

```
[C:\]type config.sys
PROTSHELL=DMPC.EXE SHELL11F.CNF SHELL11F.EXE CMD.EXE /K
OS2INIT.CMD
LIBPATH=C:\;C:\OS2;C:\OS2\INSTALL;
BUFFERS=30
DISKCACHE=64
MAXWAIT=3
MEMMAN=SWAP,MOVE
PROTECTONLY=NO
SWAPPATH=C:\
THREADS=64
SHELL=COMMAND.COM /P
BREAK=OFF
FCBS=16,8
RMSIZE=640
RUN=C:\OS2\SPOOL.EXE /D:LPT1 /O:LPT1
DEVINFO=SCR,VGA,C:\VIOTBL.DCP
DEVICE=C:\OS2\EGA.SYS

[C:\]
```

You can use EDLIN to revise the contents of the CONFIG.SYS file or any other OS/2 file saved in ASCII format. Unfortunately, EDLIN only works under DOS. Thus, you must first switch to the DOS Command Prompt to invoke EDLIN, listing the name of the file you want to edit in the command line. Then, after you have edited and saved the file, you must reboot your system for the changes to take effect. Thus, you should save any files and complete other activities in existing sessions, or you will lose some work when you reboot.

PROTSHELL Configuration Command

The PROTSHELL configuration command specifies the user interface program and OS/2 command processor that will be loaded. The standard OS/2 user interface, which is the DMPC.EXE file, requires you to specify the name of the Program Selector Configuration file (SHELL11F.CNF), the name of the Program Selector program file (SHELL11F.EXE) and the name of the OS/2 protected mode command processor

(CMD.EXE). The latter file is equivalent to DOS's COMMAND.COM and has a /C parameter similar to COMMAND.COM's /C option, which enables you to pass a command to a copy of the command processor and return to the original command processor. CMD.EXE has an additional optional parameter, /K, which is the default parameter used in the CONFIG.SYS file. This parameter passes a command to a copy of the command processor; however, it does not return to the previous command processor.

LIBPATH Configuration Command

The LIBPATH (library search path) configuration command specifies the directory or set of directories to be searched when an OS/2 program loads dynamic link libraries. The format of this command is

```
LIBPATH=[d:]path[;path. . .]
```

Unlike the PATH command, the current directory is not searched first. At installation, the LIBPATH statement placed in your CONFIG.SYS file causes the root directory, the OS2 directory, and the subdirectory INSTALL under the OS2 directory to be searched as illustrated in Listing 11.1.

BUFFERS Configuration Command

The BUFFERS configuration command is used to specify the number of 512-byte disk buffers the system will use. The format of this command is

```
BUFFERS=n
```

where *n* is a number between 1 and 100, with values greater than 100 ignored. The default value under OS/2 is 30, resulting in 15,360 bytes of storage that OS/2 uses to read and write blocks of data that do not occupy an entire sector. You can increase the speed of disk I/O operations by increasing the value specified for BUFFERS. Unfortunately, doing so involves trade-offs, because it decreases your system's RAM available to programs. You can experiment by increasing or decreasing the number of buffers. When you increase the number of buffers, you will notice a large increase in disk access speed when buffered information is repeatedly read. As an example, if you use the DIR command to list the contents of your fixed disk, often the second time you issue the command the disk activity indicator light does not illuminate. This is because a large enough disk buffer area will enable a large enough quantity of disk sectors to be kept in the disk buffer area to permit a second disk access command to read memory rather than the disk. Unfortunately, too many buffers can actually increase access time as it could take OS/2 longer to search through the disk buffer area than read the appropriate sectors from disk. Due to this, OS/2 limits the number of buffers that can be used to 100.

DISKCACHE Configuration Command

As previously mentioned in this chapter, the DISKCACHE configuration command is used to specify the number of blocks of storage that are allocated for use by the disk cache as well as for control information. The format of this command is

```
DISKCACHE=n
```

where *n* is a number between 64 and 7200 that specifies the number of 1024-byte blocks of storage to be allocated. Similar to the use of the BUFFERS command, DISKCACHE also involves a trade-off of performance versus storage, because increasing the size of your disk cache reduces the size of available storage.

MAXWAIT Configuration Command

The MAXWAIT configuration command is one of three that govern the operation of OS/2 multitasking. The other commands that influence multitasking are THREADS and PRIORITY.

MAXWAIT sets the maximum waiting time in seconds that a process can wait because of other, higher-priority processes using the processor's resources. By the use of the MAXWAIT command, you can ensure that no process waits longer than a specified period of time, after which its priority will be temporarily increased. The format of this command is

```
MAXWAIT=X
```

where *X* can be set between 1 and 255 seconds, with the default value 3 seconds.

MEMMAN Configuration Command

The MEMMAN command governs memory management options for OS/2 mode operations. Through this command you can control whether program segments are swapped to disk or relocated in memory to eliminate memory fragmentation. The latter results from OS/2's ability to subdivide a program into segments that can be placed in noncontiguous memory areas or even stored on disk and swapped to and from memory. The format of this command is

```
MEMMAN=[{SWAP   ,MOVE  }]
       [{NOSWAP ,NOMOVE}]
```

The SWAP parameter enables segment swapping, whereas NOSWAP prevents segment swapping. The MOVE parameter permits storage compaction, while NOMOVE prevents storage compaction.

The default value of MEMMAN is SWAP ,MOVE if OS/2 is started from a fixed disk.

PRIORITY Configuration Command

The PRIORITY command selects the manner in which the priority calculation in scheduling regular class threads occurs in OS/2 mode. Under OS/2, a scheduler partitions available CPU cycles among dispatchable entities called *threads.* At any given time a thread is either waiting for I/O or another event to occur, in which case it is blocked; or it is ready to execute or is being executed. Regular class threads have their priority dynamically varied by OS/2, which is the default for this command. This setting causes the priority of each thread to vary, based on the activity of the other threads in the system. If you do not want OS/2 to vary the priority of threads as they are running,

you can set the command's value to ABSOLUTE. Due to this, the format of the command is

```
PRIORITY= {DYNAMIC }
          {ABSOLUTE}
```

PROTECTONLY Configuration Command

This configuration command either selects OS/2 as the only mode or enables you to have both OS/2 and DOS operating modes. The format of this command is

```
PROTECTONLY= {NO }
             {YES}
```

where NO results in both DOS and OS/2 operating modes, whereas YES results in only an OS/2 mode. A few years or perhaps a decade from now, when DOS programs have run their course, you may want to change the default of NO to YES. Until then, you will probably want to run both modes by accepting the default value in the CONFIG.SYS file.

SWAPPATH Configuration Command

The SWAPPATH configuration command defines the location of the file used by OS/2 that keeps track of storage segments that are swapped in and out of memory. The format of this command is

```
SWAPPATH=[d:][path]
```

The default setting for this command is C:\SWAP. Because the swap file requires a minimum of 512K bytes of storage, you must have that amount of available storage on the swapping drive. If you use the MEMMAN command to specify NOSWAP, you can obtain an additional 512K bytes of storage to use for other purposes.

THREADS Configuration Command

The THREADS command specifies the number of independent actions that can simultaneously exist in your system. The format of this command is

```
THREADS=X
```

where *X* is a number between 32 and 255. The default value of 64 includes an allowance for 24 threads used by OS/2 and 40 for application programs.

SHELL Configuration Command

The SHELL command replaces the DOS command processor with another command processor. This provides software developers with the ability to develop customized versions of the DOS command processor and to easily invoke their use. The format of this command is

```
SHELL=[d:][path]filename[arguments]
```

The default value for SHELL causes the COMMAND.COM command processor provided with OS/2 to be used for DOS mode. The /P parameter used in the command line in the CONFIG.SYS file causes the DOS processor to permanently reside in storage, enabling it to always be selected from the Program Selector.

RMSIZE Configuration Command

The RMSIZE command is only applicable for DOS mode and represents a new configuration command introduced with OS/2. This command enables you to specify the highest storage address used by DOS. The format of this command is

```
RMSIZE=X
```

where *X* is a number from 0 to 640 that represents a multiple of 1024 bytes of memory. The default value used by OS/2 is 640, which results in the maximum amount of memory DOS can use. If your DOS programs require less memory, you can assign more memory for protected mode operations by using the RMSIZE command to reduce or eliminate memory assigned for the DOS mode. Thus, changing the command to RMSIZE=384 would allocate an additional 256K bytes of memory to OS/2 mode.

Batch Commands

All batch file commands that operate under DOS work the same under OS/2. When OS/2 was introduced, three additional batch commands were added that are only applicable for OS/2 mode operations—ENDLOCAL, EXTPROC, and SETLOCAL. This section examines the use of batch files under OS/2, their similarities and differences with respect to DOS batch files, and the use of the three new batch commands.

Extensions

Because there are some OS/2 commands that only operate in OS/2 mode, whereas some DOS commands only operate in DOS mode, you can use different extensions to denote the mode a particular batch file was programmed for. A batch file with the extension .BAT only operates in DOS mode. Similarly, a batch file with the extension .CMD only operates in OS/2 mode.

Exercise caution in constructing batch files: The commands included in a batch file govern their operation. Thus, regardless of the extension used, you should ensure that you do not put a command that only works in OS/2 mode into a batch file whose extension is for DOS mode.

Special Batch Files

When you install OS/2, the installation process results in the creation of two special batch files—OS2INIT.CMD and AUTOEXEC.BAT. Both of these files are placed into your root directory and contain the default search paths to external commands and data files for each operating system mode.

The OS2INIT.CMD file contains the following two commands.

```
PATH C:\;C:\OS2;C:\OS2\INSTALL;
DPATH C:\;C:\OS2;C:\OS2\INSTALL;
```

The AUTOEXEC.BAT file contains the following statement:

```
PATH C:\;C:\OS2;
```

The OS2INIT.CMD is a special type of file similar to an AUTOEXEC.BAT file. That is, each time you start an OS/2 session the OS2INIT.CMD file, if it exists, is executed. Similarly, the first time you select the DOS command prompt from the Program Selector, the AUTOEXEC.BAT file, if it exists, is executed.

Because the OS2INIT.CMD and AUTOEXEC.BAT files define search paths, you probably prefer to modify them to include paths to your application programs and data files they use or to supplement their use. You can modify them with EDLIN; however, you will have to do so in DOS mode, because EDLIN only operates in that mode. Concerning supplementing the paths in the batch files, you can enter a PATH or DPATH command during a session to temporarily change or extend an existing path.

STARTUP.CMD File

The STARTUP.CMD file is a special batch file you can use to start programs in OS/2 sessions when OS/2 is initialized. This file must be located in the root directory of your startup drive and can be considered as OS/2's equivalent to DOS's AUTOEXEC.BAT batch file. STARTUP.CMD is more powerful, because it permits you to start programs that execute concurrently in additional OS/2 sessions, whereas programs can only execute consecutively in an AUTOEXEC.BAT file.

The sample batch file shown here illustrates the contents of a STARTUP.CMD file that would initiate three OS/2 sessions.

```
PATH C:\OS2;\SPJ;C:\MM
DPATH C:\;C:\OS2;\SPJ;C:\MM
START SPJ
START MM
```

When you start OS/2, the STARTUP.CMD file initiates a command processor in a session that will be displayed on your screen. This session processes the statements in the STARTUP.CMD file. Next, paths for external commands, programs, and data files are set. Then, the first START command starts a second session that runs the SPJ program. Finally, the second START command initiates a third session that runs the MM program.

New Batch Commands

As previously mentioned, OS/2 includes three new batch commands that are only applicable to OS/2 mode.

SETLOCAL (Environment) Command

The SETLOCAL command saves the current drive, directory, and system environment variables as well as defines local variables for the batch file. You can later restore

previously saved settings using the ENDLOCAL command within the batch file or when the batch file ends. The format of this command is

```
SETLOCAL
```

ENDLOCAL (Environment) Command

The ENDLOCAL command restores the drive, directory, and environment variables that were in effect before a SETLOCAL command was executed. The format of this command is

```
ENDLOCAL
```

The following list illustrates the use of the SETLOCAL and ENDLOCAL commands. Here the left column shows the statements in the batch file, whereas the right column indicates the resulting display when each command in the file is executed.

Batch File Entry	Command Prompt Display
`PATH`	`[C:\]PATH`
	`PATH=C:\OS2`
`SETLOCAL`	`[C:\]SETLOCAL`
`PATH C: OS2 SPJ`	`[C:\]PATH C: \OS2\SPJ`
`ENDLOCAL`	
`PATH`	`[C:\]PATH`
	`PATH=C:\OS2`

EXTPROC (External Processor) Command

The EXTPROC command defines an external batch processor for an OS/2 mode batch file. This command is useful if you want to substitute a third-party batch processor for the OS/2 batch processor that may include additional batch commands you can use for creating batch files or processing commercially sold batch files. The format of this command is

```
EXTPROC [d:][path]filename.ext[arguments]
```

where the arguments are optional parameters that can be passed to the new batch processor.

12 // OS/2 Presentation Manager

The OS/2 Presentation Manager provides a graphic user interface for IBM's new operating system and represents the first implementation of IBM's System Application Architecture (SAA). Under SAA, a consistent set of program and user interfaces to applications can be run on all IBM computers. Thus, the Presentation Manager can be considered to represent the "look and feel" of both existing application program interfaces and those to be developed to operate under OS/2's graphic operating system environment.

The history of the development of graphic user interfaces can be traced to Xerox Corporation's Palo Alto Research Center where, in the early 1970s, software developers used graphic representations of such office tools as filing cabinets to represent storage, folders to represent groups of files, and a wastebasket for clearing objects from the screen. Although Xerox attempted to commercialize its graphic user interface, actual commercial success in this area resulted from Apple Computer Company's second graphic user interface based personal computer, the Macintosh. Apple's first graphic user interface-based personal computer called Lisa, like Xerox's initial offering, was felt to be overpriced when it reached the market.

With the growing acceptance of the Macintosh graphic user interface, it was not long until several graphic interfaces were developed for use on DOS machines, such as the original IBM PC series and the family of PS/2 personal computers. Examples of graphic user interfaces developed for DOS systems include Microsoft Corporation's Windows and Digital Research's GEM. Similar to those operating environments, the OS/2 Presentation Manager has a multiple windowing capability and pull-down menus, as well as some extras, such as an internal help facility. The Presentation Manager also can manipulate several computing tasks at one time, due to the multitasking capability of the operating system.

Installation and Initial Operation

Similar to the installation of DOS 4.0 and OS/2 Standard Edition, IBM simplified the installation of the Presentation Manager, providing an installation diskette that guides you through the sequence of operations required to install the operating system. By inserting the IBM Operating System/2 Installation Diskette in drive A and powering-on your computer or performing a system reset, you are guided through the installation process by the display of a series of easy-to-follow screens. Screen displays prompt you

when to insert and remove each of the OS/2 diskettes, as well as how to select the required country, keyboard, serial device, and mouse.

Currently, OS/2 supports the IBM Mouse, serial version 039-099 and 039-199 of the Microsoft Mouse, PC Mouse Systems' Mouse, and the Visi-On Mouse. Of course, you can also select None from the Select Mouse screen; however, unlike DOS 4.0—which can be as easily manipulated from the keyboard as with a mouse—it is advisable to use a mouse with the Presentation Manager. This is because the Presentation Manager graphical windowing environment screen displays typically have a higher density of on-screen information than displayed in character-oriented screens or the single-tasking DOS 4.0 screen. Thus, the use of a mouse greatly facilitates Presentation Manager operations, especially when multiple windows are displayed and you want to switch between them.

Because the Presentation Manager supports both keyboard entries and the mouse, you can mix your operations to take advantage of the capabilities of each as well as adjust their use to correspond to a level of use you are comfortable with. In general, using a mouse helps you browse through sequences of activities faster than you could using keyboard command entries. For certain operations, the system provides keyboard shortcut methods of performing an operation, especially as you gain experience in the use of OS/2 commands.

Working with Windows

After OS/2 Version 1.1 is installed, whenever your computer is powered-on or a system reset is performed, the Start Programs window is displayed, centered in the middle of your screen as illustrated in Figure 12.1. Initially, this window has four entries. Other OS/2 programs can be listed in this window once they are installed—a process discussed later in this chapter. From this window you can use the File System entry to view your file and directory structure, run a program that provides an overview of OS/2 (Introducing OS/2), or issue OS/2 commands on either a full-screen basis (OS/2 full screen command prompt) or from within a window (OS/2 window command prompt). Once either OS/2 command prompt entry is selected, the default drive and current directory are displayed in brackets, similar to OS/2 Standard Edition 1.0. Thus, if drive C is the default drive and you are located at its root directory, the OS/2 Presentation Manager command prompt would be displayed as [C:\]. Prior to analyzing the entries in Figure 12.1, you should take a look at the parts of an OS/2 window as well as some of the icons you can display at the bottom of your screen.

Parts of a Window

Figure 12.2 illustrates the various parts of a Presentation Manager window. This section discusses each of the parts of a window and how you can select them by the use of a mouse or, if permitted, by the use of the keyboard. Following convention, the term *click* denotes pressing and releasing a mouse button, *double-click* refers to pressing and releasing a mouse button twice, and *drag* means moving the mouse while you hold

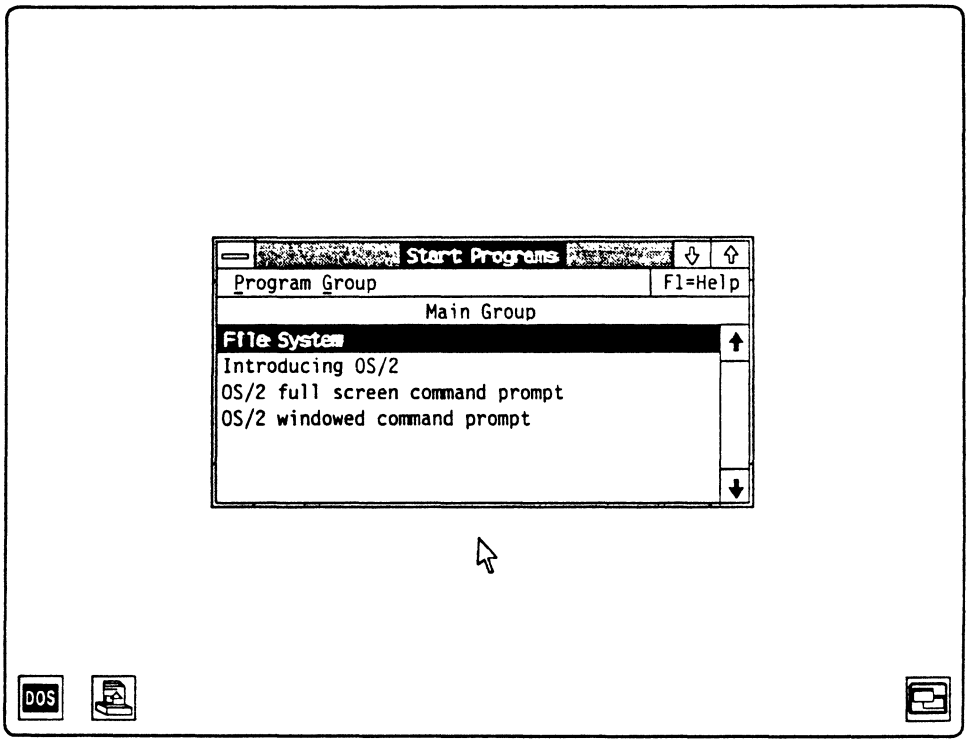


Figure 12.1
Start Programs, the Initial Presentation Manager Window

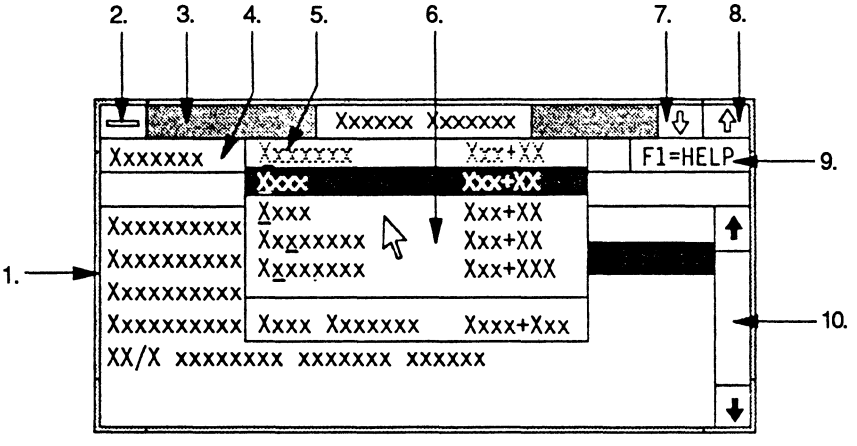


Figure 12.2
Presentation Manager Window Parts

Legend:

1. Border
2. System Menu Icon
3. Title Bar
4. Action Bar
5. Action Bar Choice (Keyword)
6. Pull-Down Menu
7. Minimize Icon
8. Maximize Icon
9. F1=Help (Help Facility)
10. Scroll Bar

down the mouse button. Similarly, the term *select* denotes the process of marking or highlighting an item by a keyboard entry or mouse operation.

Border

The border (which is 1 in Figure 12.2) is the boundary of a window. When the border is highlighted, the window is active and currently in use. This means that keyboard entries or mouse operations on the window affect the window. If two or more windows are displayed, the topmost window normally is the active window. Although you cannot select a border using a keyboard entry, the mouse pointer changes its shape into a double arrow when it crosses a border for window sizing purposes. Then you can click on the window to make it the current window. If you are using the keyboard, you can press the **Alt** and **Tab** keys to switch to the next window.

System Menu Icon

The system menu icon (2) located in the upper left corner of the window can be used to display the system menu. From that menu, you can move or size a window. In addition, depending on the window in use, you may be able to use the system menu icon to switch to another window, change the fonts of a program running in the window, or close a window.

Figure 12.3 illustrates the system menu of the Start Programs window. This menu is selected with a mouse by clicking on the system menu icon or by pressing **Shift+Esc** or **Alt+Space bar**. Note that you can select an item in the menu by moving the pointer to the appropriate entry and clicking on it, by pressing the letter underlined in the entry, or by pressing the indicated key combination. In fact, you can bypass the system menu and simply press the appropriate key combination to initiate the selection of a desired action. Also note that a window whose size is minimized is displayed as an icon at the bottom of the screen. Here an icon represents an application running under OS/2 without an output window. Once a window is minimized, it can be restored to its previous size by pressing **Alt+Esc** until the system menu for the icon is displayed. Then, you can click on Restore or press **Alt+F5**.

Title Bar

The title bar (3) identifies a window as it contains the title of the window. You can move a window with a mouse by first selecting the title bar. This is accomplished by clicking anywhere on the title bar and then dragging the window to a new location on the screen. You cannot select a title bar using the keyboard.

Action Bar

The action bar (4) contains keywords used to display the options of functions available for selection from the current window. In Figure 12.1, Program and Group are the keywords that can be selected. If you are using a mouse, you can select a keyword by clicking on the word. If you are using the keyboard, press **F10** key to select the action bar. This highlights the first keyword in reverse video.

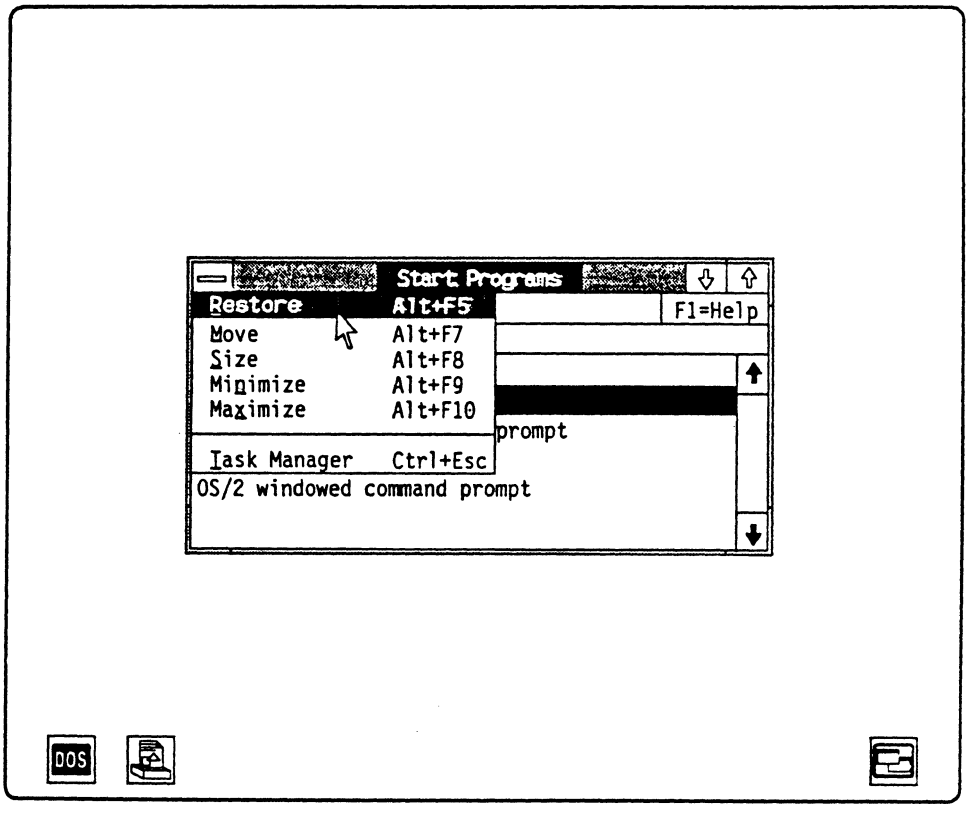


Figure 12.3
System Menu of the Start Programs Window

Action Bar Choice

The keyword or keywords (5) in the action bar are action bar choices. Once the action bar is selected, you can either press the key of a number or letter that is underlined in the action bar choice or press the **Left** or **Right arrow** key to position the highlight inverse video bar over the keyword. For the latter operation, you must then press **Down arrow** or **Enter** to display a pull-down menu containing options of functions available for selection. Thus, if the Start Programs window shown in Figure 12.1 is the active window, you could press **F10** to select the keyword `Program` in the action bar and enter **G** to display the group pull-down menu.

Pull-Down Menu

A pull-down menu (6) lists options for tasks you can perform from the current window. Initially, an inverse video highlight bar appears over the first selectable option in the pull-down menu. You can select an option from a pull-down menu with a mouse by clicking on the option or from the keyboard by either pressing the **Down** or **Up arrow** key to position the highlight bar over the option and then pressing **Enter**. As an alternative, you can select any displayed option in a pull-down menu—unless it is

"dimmed"—by typing the letter or number that is underlined in the option. If the option is selectable, it will appear in black text. If the option cannot be selected at that particular time, it will appear dimmed in gray text.

Minimize Icon

The down arrow at the upper right corner of the window is known as the *minimize icon* (7). By selecting it, you reduce the current window's size to a small icon that will be positioned at the bottom of your display. Check the bottom of Figure 12.1; notice the three icons there. Each icon represents a program running in a window that has been minimized. The first icon at the bottom of the screen labeled DOS represents a DOS full-screen command prompt window. Thus, to start a DOS program, you can simply double-click on this icon. If you are not using a mouse, you cannot select icons that represent minimized windows; however, you can initiate a set of keyboard actions that are the equivalent of double-clicking on an icon; those actions are explained later in this chapter. The other two icons at the bottom of Figure 12.1 represent the Spooler Queue Manager and Task Manager windows, respectively. Both windows are discussed later in this chapter. You can either click on the minimize icon or press **Alt+F9** from the keyboard to minimize the size of a window.

Maximize Icon

The *maximize icon* (8) enlarges the size of the current window to fill the entire screen. You can click on the maximize icon or press **Alt+F10** to maximize the size of a window.

F1=Help

The help facility (9) built into OS/2 can be selected by either clicking on F1=Help or pressing **F1**. Once selected, a window containing general information is displayed.

Scroll Bar

The Scroll Bar (10) enables you to view information that exceeds the display capability of the window. Although you can only select the scroll bar with a mouse, you can use the **Down** and **Up arrow**, **PgDn**, **PgUp**, **F8**, or **F7** to scroll information in a window. When there is more information than can be viewed in a window at one time, use the scroll bar's slider box to display text hidden above or below the current display, similar to the illustration presented in Figure 12.4. You can rapidly scroll several lines of text by moving the mouse pointer on the slider box and dragging the mouse in the direction you want to scroll information. To scroll one page at a time, you should click on a part of the scroll bar above or below the slider box. If you click on a part of the scroll bar above the slider box, the information scrolls one page up, whereas clicking on a part of the scroll bar below the slider box scrolls the information downward one page.

Now that you have examined the parts of a window, you are ready to explore the use of two key OS/2 windows—Start Programs and Task Manager.

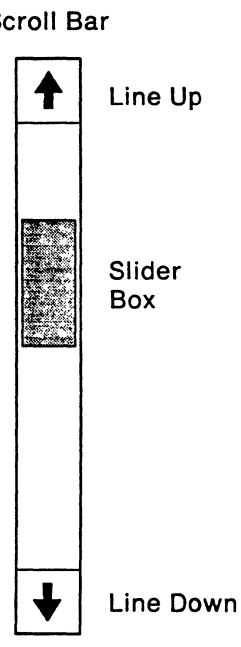


Figure 12.4 Slider Box

The Start Programs Window

The Start Programs window illustrated in Figure 12.1 is the initial window displayed when the OS/2 Presentation Manager is placed into operation. The first time you use OS/2 Version 1.1—the more formal name for the OS/2 Standard Edition Presentation Manager—there are four selectable entries in the window. These entries enable you to go directly to the file system, execute the OS/2 tutorial, or issue OS/2 commands in a full-screen or in a windowed environment.

To view files and directories, you can double-click on the File System entry or select it and press **Enter**. This action results in the display of a Directory Tree window that shows the structure of your computer's directory to include subdirectories and files similar to those provided under DOS 4.0.

Program Keyword

By pressing **F10** or clicking on the action bar, you select the first keyword. When you work with the Start Programs window, either action selects the Program keyword. If you then press the Down arrow key or click on `Program`, its pull-down menu is displayed. Table 12.1 lists the items you can select from the Program pull-down menu.

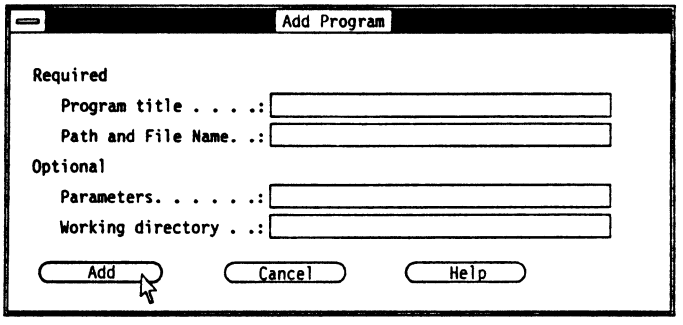
Program titles are organized into groups; only one group at a time is displayed in the Start Programs window. Initially, the Main Group illustrated in Figure 12.1 is one of two groups contained in the Presentation Manager. It contains four programs. The second group, Utility Programs, is discussed later in this chapter.

If you select Start from the Program pull-down menu, whatever program was previously highlighted in the Start Program window is started. In examining Figure 12.1,

Table 12.1
Program Pull-Down Action Items

Action Item	Operation
Start	Starts a program.
Add	Adds a new program title.
Change	Displays the program title so you can change it.
Delete	Deletes a program title.
Copy	Copies a program title.
Minimize or run	Converts the Start Programs window into an icon when you start a program.

Figure 12.5
The Add Program Window



note that File System is highlighted. Thus, selecting Start from the Program pull-down menu causes the program that displays your computer's directory structure to run.

Because the Main Group is currently displayed in the Start Program window, all Program pull-down menu operations reference that group. As an example of working with the Main Group, assume you wish to add an executable OS/2 program named WORD.EXE located in the subdirectory WP under the root directory of drive C to the Main Group. Also assume you want the title "Word Processor" to appear in the Main Group. To accomplish this you first select Add from the Program pull-down menu. This displays the Add Program window illustrated in Figure 12.5. To add the word-processing program to the Main Group, you first type **Word Processor** into the Program title box, illustrated in Figure 12.5. Next, enter **C:\WP\WORD.EXE** into the Path and File Name box. Depending on the operational design of the program, you may enter optional parameters that govern the execution of the program. Similarly, if the program uses files in a working directory, you would enter the location of that directory in the working directory box. Once the preceding operations are completed, you would either click on Add or press **Enter**.

Note that when using the keyboard, you can press the **Tab** key or arrow keys to rapidly move a vertical bar that indicates the box you are working with. This vertical bar jumps from box to box or into each of the three selection buttons (Add, Cancel, and Help) at the bottom of the Add Program window. Once the Add button is selected, Word Processor is added to the Main Group, as illustrated in Figure 12.6.

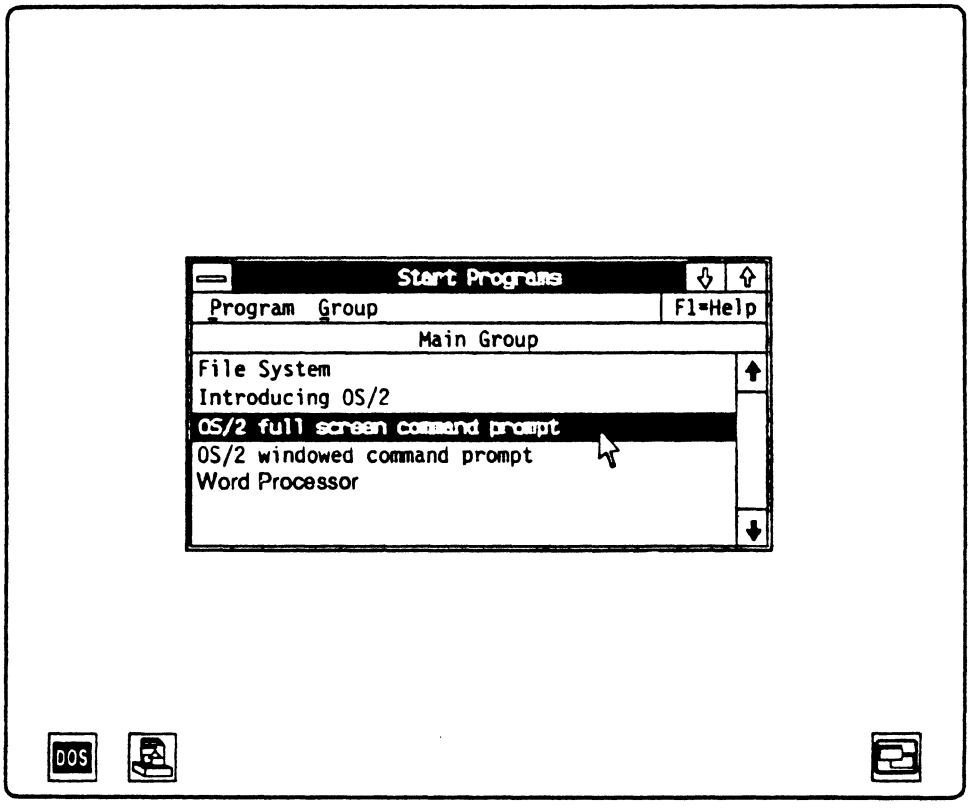


Figure 12.6
Word Processor Program Added to Main Group

Table 12.2
Group Pull-Down Action Items

Action Item	Operation
Add	Creates a new pull-down group and adds its name to the list in the group pull-down menu.
Delete	Removes an empty program group.
Rename	Changes the name of a program group.
1. Main Group	Displays the Main Group.
2. Utility Programs	Displays the names of utility programs.

Group Keyword

The Group pull-down menu options create, modify, or delete program groups as well as switch to a particular group of programs. Table 12.2 lists the action items initially contained in the Group pull-down menu and the operations resulting from their selection.

The Group pull-down menu groups programs together in a common window. By doing so it helps you to locate and run programs. For example, if you have several

word-processing programs, you might want to group them under the title, "Word Processors."

Once the Group pull-down menu is displayed, you can either click on Add or select it and press **Enter**. Either action results in the display of the Add a Group window, permitting you to type the name you want assigned to a new group. Similar to the Add Program window, clicking or selecting the Add button at the bottom of the Add a Group window invokes the option. The new group is entered and can be verified by viewing the Group pull-down menu, where it would be placed under the Utility Programs entry on that menu.

When the Group pull-down menu is first viewed, it contains two numbered entries—Main Group and Utility Programs. Selecting Main Group displays the Main Group in the Start Programs window, whereas selecting Utility Programs causes a Utility Programs window to be displayed. The latter window initially contains four selectable entries—Control Panel, Disk Information, Format Diskette, and OS/2 System Editor. By selecting the Control Panel, you can set or reset the time and date, as well as define the cursor blink and mouse double-click rates. Disk information invokes the CHKDSK program, while Format diskette invokes the FORMAT program. The final entry in the Utility Programs window enables you to select the OS/2 System Editor, a new feature added to OS/2 with the introduction of the Presentation Manager.

To add programs to a group is a simple process. First, display the Group pull-down menu and click on the desired group or select it. Then display the Program pull-down menu and click on or select Add or Copy to add or copy a program title to the group you are working with.

Start Programs System Menu

The Start Programs System Menu pull-down can be selected by clicking on the system menu icon, pressing **Shift+Esc** or **Alt+Space bar** or pressing **F10** and the **Right arrow** key two times. The latter action causes the keyword `Program` to be highlighted by the F10 key, whereas pressing **Right arrow** twice moves the highlight bar over `Group` and then to the system menu icon. Once the system menu icon is selected, press **Down arrow** to display the System Menu of the Start Programs window. This menu's options manipulate the current window size and its screen location, as well as switch to the Task Manager window. Table 12.3 lists the options in the Start Programs System Menu

Table 12.3
Start Programs System Menu Action Items

Action Item	Keyboard Entry to Invoke
Restore	Alt+F5
Move	Alt+F7
Size	Alt+F8
Minimize	Alt+F9
Maximize	Alt+F10
Task Manager	Ctrl+Esc

and the key combinations you can enter to rapidly invoke those actions without displaying the menu.

The Restore function restores a window to its size prior to a minimize or maximize operation. The Move and Size options cause a grayed outline to overlay the border of the window. Once you select either function, you can use the arrow keys to move or size the window. The last entry in the Start Programs System Menu, Task Manager, switches to the Task Manager. To do so, you can select the Task Manager option from the System Menu and press **Enter**, type **T** when the System Menu is displayed, or press **Ctrl+Esc**. Because the Task Manager may be included in the System Menu of other windows, its display is not restricted to the Start Programs System Menu.

The Task Manager

The Task Manager window switches between running programs. When first selected from the Start Programs window, the Task Manager window contains three entries—DOS Command Prompt, Start Programs, and Spooler Queue Manager. Figure 12.7 illustrates the initial Task Manager window.

From the Task Manager window, you can either double-click on a program title you want to switch to, or you can select it and press **Enter**. As an alternative, you can also select Switch to from the Task pull-down menu, discussed later in this section.

Selecting the DOS Command Prompt entry in the Task Manager window displays a full-screen window from which you can run one DOS program at a time. When this window is displayed, the DOS prompt indicates the default drive and current directory. Thus, if drive C is the default drive and the root directory is the current directory, the prompt is indicated by C:\>. The top line in the DOS Command Prompt window will note Ctrl+Esc=Task Manager and Type HELP=help and will not scroll, providing a reference on how to return to the Task Manager or how to invoke the help facility.

Selecting Start Programs in the Task Manager window returns you to the Start Programs window. The third entry in the Task Manager window, Spooler Queue Manager, invokes the Spooler Queue Manager program window, which enables you to check the status of print jobs, prioritize jobs to print in a specified order, reprint jobs, cancel jobs you no longer want printed, or place a job on hold until you release it. If you

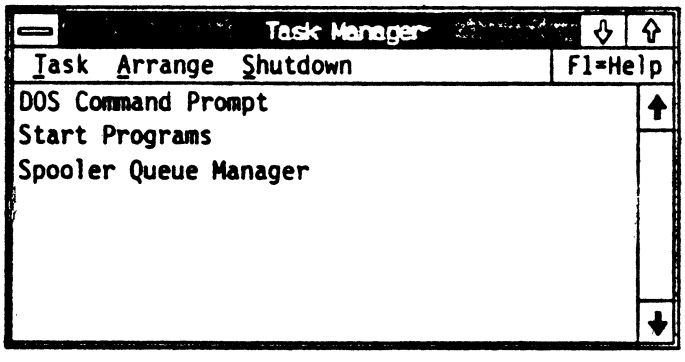


Figure 12.7
Initial Task Manager Window

previously began to run one or more OS/2 programs, those programs also are listed in the Task Manager window.

Task Pull-Down Menu

The Task pull-down menu in the Task Manager window contains three action items you can select—`Switch to`, `Close`, and `Minimize after use`. Selecting `Switch to` activates the program previously selected in the Task Manager window, so the topmost window contains the program. The `Close` option enables you to end a program, saving your data and closing any open files. Selecting `Close` has the same effect as selecting Exit from within a program. To close a program, first select the desired program from the list of running programs. Then, selecting `Close` causes the program to terminate.

The `Minimize after use` option in the Task pull-down menu minimizes the Task Manager window to an icon after the option is selected. When selected, a check mark prefixes the item. To remove the check mark, simply select `Minimize after use` again.

Arrange Pull-Down

The Arrange pull-down menu in the Task Manager window contains two options—`Cascade` and `Tile`. Both entries arrange programs running in windows into one of two orderly predefined patterns. Figure 12.8 illustrates the result of selecting each entry.

Shutdown Pull-Down

The third pull-down menu in the Task Manager, Shutdown, ends all OS/2 programs that are running, whether or not they are running in windows. To accomplish this, select the `Shutdown now` option. The only other entry in this pull-down is `Resume Task Manager`.

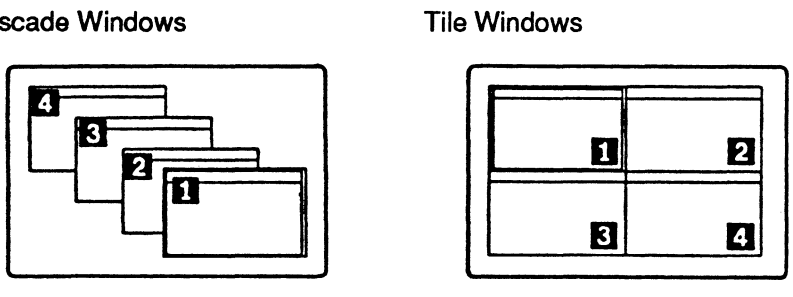


Figure 12.8 Arrange Pull-Down Results

13 / Fundamental Communications Concepts

The first of four chapters that cover data communications, this chapter concerns fundamental communications concepts that provide the foundation for mastering the operation and use of different types of communications adapter boards. The remaining three chapters use the information presented in this chapter in covering System/3X and 3270 networking, local area networks, and reviewing networking strategies one can consider in using a PS/2 to communicate with other personal computers, mainframe computers, and information utilities.

Three Elements for Communications

To transmit information between two locations, you need a transmitter, a receiver, and a transmission medium that provides a path between the transmitter and the receiver. In addition to transmitting signals, a transmitter must be capable of translating information from a form created by humans or machines into a signal suitable for transmission over the transmission medium. The transmission medium provides a path to convey the information to the receiver without introducing a prohibitive amount of signal distortion that could change the meaning of the transmitted signal. The receiver then converts the signal from its transmitted form into a form intelligible to humans or machines.

Whereas the transmission of data may appear to be a simple process, many factors govern the success or failure of a communications session. In addition, the performance and economics associated with the use of an IBM PS/2 in a data communications environment can vary considerably, depending on numerous variables. Such variables can include

- the type of communications hardware and software used with one's personal computer
- the transmission medium employed
- the method by which the personal computer is connected to other network devices that may be required to integrate the computer into an existing network

This chapter reviews the fundamental concepts associated with data communications. This provides a background in data communications concepts for examining the various options you can consider in a networking environment, as presented later in this book.

Line Connections

Three basic types of line connections are available to connect personal computers to other computers: dedicated, switched, and leased lines.

A *dedicated line* is similar to a leased line in that the personal computer is always connected to the device on the distant end. Transmission always occurs on the same path, and, if required, the line can be easily tuned to increase transmission performance. The key difference between a dedicated and a leased line is that a dedicated line refers to a transmission medium internal to a user's facility, where the customer has the right of way for cable laying, whereas a leased line provides an interconnection between separate facilities. The term *facility* is usually employed to denote a building, office, or industrial plant. Dedicated lines are also referenced as *direct connect lines* and normally link a personal computer, terminal, or business machine on a direct path through the facility to another personal computer, terminal, or computer located at that facility. The dedicated line can be a wire conductor installed by the employees of a company or by the computer manufacturer's personnel, or it can be a local line installed by the telephone company. Normally, the only cost associated with a dedicated line in addition to its installation cost is the cost of the cable required to connect the devices that are to communicate with one another.

A *leased line* is commonly called a *private line* and is obtained from a communications company to provide a transmission medium between two facilities that can be in separate buildings in one city or in distant cities. In addition to a one time installation charge, the communications carrier normally bills the user on a monthly basis for the leased line, with the cost of the line usually based on the distance between the locations connected by the line.

A *switched line,* often referred to as a dial-up line, permits contact with all parties having access to the *public switched telephone network (PSTN).* If the operator of a personal computer wants access to another computer, he or she dials the telephone number of the telephone line, which in turn is connected to the other computer. In using switched or dial-up transmission, telephone company switching centers establish a connection between the dialing party and the dialed party. After the connection is set up, the devices at each end of the line conduct their communications. When communications are completed, the switching centers disconnect the path that was established for the connection and restore all paths so they become available for other connections.

The cost of a call on the PSTN is based on many factors, including the time of day when the call was made, the distance between called and calling parties, the duration of the call, and whether operator assistance was required in placing the call. Direct dial calls made from a residence or business telephone without operator assistance are billed at a lower rate than calls requiring operator assistance. In addition, most telephone companies have three categories of rates: *weekday, evening,* and *night and weekend.* Calls made between 8 A.M. and 5 P.M. Monday through Friday are normally billed at a "Weekday" rate, while calls between 5 P.M. and 11 P.M. on weekdays are usually billed at an evening rate, which reflects a discount of approximately 25 percent from the weekday rate. The last rate category, night and weekend, is applicable to calls made between 11 P.M. and 8 A.M. on weekdays as well as any time on weekends and holidays.

However, some communications carriers may have a rate change at 5 P.M. on Sunday. Calls during the weekend rate period are usually discounted 50 percent from the weekday rate.

Table 13.1 contains a sample PSTN rate table that is included for illustrative purposes but that you should not use to determine the actual cost of a PSTN call. The cost of intrastate calls by state and charges for interstate calls vary. In addition, the cost of using different communications carriers to place a call between similar locations typically varies. Obtain a current rate schedule of the vendor you plan to use to determine or project the cost of using PSTN facilities.

Cost, speed of transmission, and degradation of transmission are the primary factors used in the selection process between leased and switched lines. As an example of the economics associated with PSTN and leased line usage, assume a personal computer located 50 miles from a mainframe communicates between 8 A.M. and 5 P.M. with the mainframe once each business day for a period of 30 minutes. Using the data in Table 13.1, each call costs .31 × 1 + (.19 × 29), or $5.82. Assuming there are 22 working days each month, the monthly PSTN cost for communications between the PS/2 and the mainframe would be $5.82 × 22, or $128.04. If the monthly cost of a leased line between the two locations is $250, it is obviously less expensive to use the PSTN for communications. Suppose the communications lengthen to 2 hours per day. Then, from Table 13.1, the cost per call becomes .31 × 1 + (.19 × 119), or $22.92. Again assuming 22 workdays per month, the monthly PSTN charge would increase to $504.24, making the leased line more economical. Thus, if data communications requirements to a mainframe computer involve occasional random contact from a number of personal computers and terminals at different locations and each call is of short duration, dial-up service is normally the least expensive. If a large transmission volume occurs between a personal computer and another computer, leased lines are usually installed between the two devices.

Because a leased line is fixed in its routing and the suppliers know the route, the line can be *conditioned* to reduce errors in transmission as well as permit ease in determining the location of error conditions. Normally, switched circuits are used for transmission at speeds up to 9600 bits per second (bps); however, in certain situations data rates as high as 19,200 bps are achievable when transmission on the PSTN occurs through telephone company offices equipped with modern electronic switches.

Table 13.1
Sample PSTN Rate Table

Mileage Between Locations	(Cost per Minute in Cents)					
	Weekend		Evening		Night and Weekend	
	First Min.	Each Add'l Minute	First Min.	Each Add'l Minute	First Min.	Each Add'l Minute
1–100	.31	.19	.23	.15	.15	.10
101–200	.35	.23	.26	.18	.17	.12
201–400	.48	.30	.36	.23	.24	.15

Some of the limiting factors involved in determining the type of line to use for transmission between personal computers and other computers are listed in Table 13.2.

Types of Service and Transmission Devices

Digital devices, which include terminals, mainframe computers, and personal computers, transmit data as unipolar digital signals, as indicated in Figure 13.1A. Here, *unipolar* refers to the fact that marks or binary 1's are represented by a positive voltage (one pole), whereas spaces or binary 0's are represented by no voltage. When the distance between a personal computer and another computer is relatively short, transmissions of digital information between the two devices are managed by cabling the devices together. As the distance between the two devices increases, the pulses of the digital signals become distorted due to the resistance, inductance, and capacitance of the cable used as a transmission medium. At a certain distance between the two devices the pulses of the digital data will distort, so they become unrecognizable by the receiver (this distortion is shown in Figure 13.1B). To extend the transmission distance between devices, specialized equipment must be employed, with the type of equipment used dependent on the type of transmission medium employed.

Basically, data is transmitted in a digital or analog form. To transmit data long distances in digital form requires *repeaters* to be placed on the line at selected intervals to reconstruct the digital signals. The repeater is a device that essentially scans the line looking for the occurrence of a pulse and then regenerates the pulse into its original form. Thus, another name for the repeater is a *data regenerator.* As illustrated in

Table 13.2
Line Selection Guide

Line Type	Distance Between Transmission Points	Speed of Transmission	Use for Transmission
Dedicated (direct connect)	Local	Limited by conductor	Short or long duration
Switched (dial-up)	Limited by telephone access availability	Normally less than 9600 bps	Short-duration transmission
Leased (private)	Limited by telephone company availability	Limited by type of facility	Long duration or short duration calls

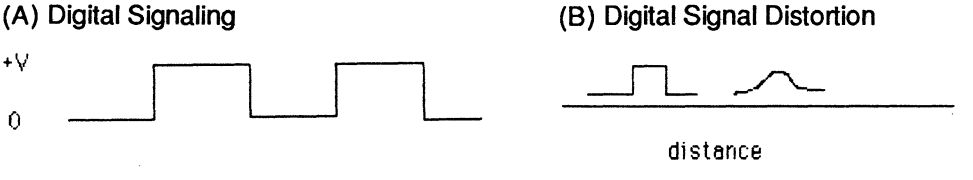


Figure 13.1
Digital Signaling and Digital Signal Distortion

Figure 13.2, a repeater extends the communications distance among terminal devices, including personal computers and mainframe computers or other business machines.

Unipolar signaling results in a DC voltage buildup when it is transmitted over long distances, so digital networks convert unipolar signals to a modified bipolar format for transmission on this type of network. This requires the installation at each end of the circuit of a device known as a *digital service unit (DSU)*. Figure 13.3 shows the use of DSUs for transmission of data on a digital network. Later this chapter examines digital facilities in more detail.

Modems

Because telephone lines were originally designed to carry analog or voice signals, the digital signals transmitted from a terminal to another digital device must be converted into a signal that is acceptable for transmission by the telephone line. To effect transmission between distant points, a data set or modem is used. *Modem* is a contraction of the compound term *modulator-demodulator* and is an electronic device that converts the digital signals generated by computers and terminal devices into analog tones for transmission over telephone network analog facilities. At the receiving end, a similar device accepts the transmitted tones, reconverts them to digital signals, and delivers these signals to the connected device.

Signal conversion by modems is illustrated in Figure 13.4. This figure shows the interrelationships of personal computers, mainframe computers, and transmission lines when analog transmission service is used. Both leased lines and switched lines employ analog service; therefore, modems can be used for transmission of data over both types

Figure 13.2
Transmitting Data in Digital Format

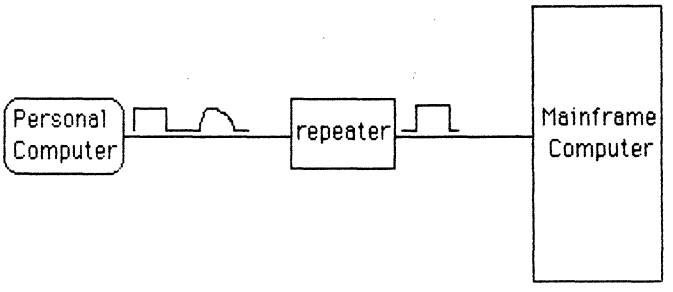
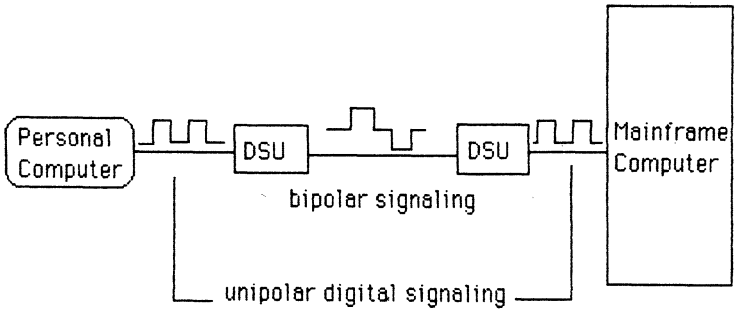


Figure 13.3
Transmitting Data on a Digital Network



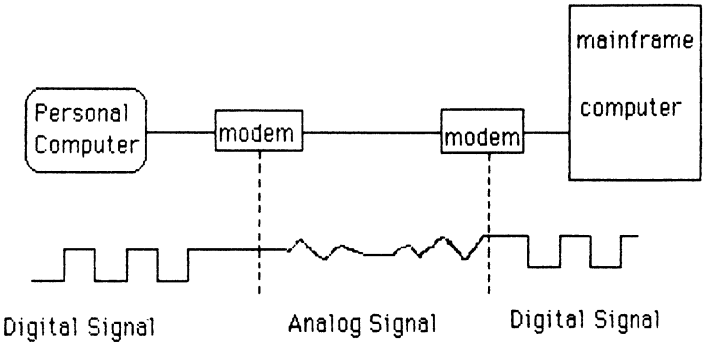


Figure 13.4
Signal Conversion Performed by Modems

of analog line connections. Although an analog transmission medium for a transmission path between modems can be a direct connect, leased, or switched line, modems are directly connected (hard wired) to direct connect and leased lines, whereas they are interfaced to a switched facility. Thus, a terminal user can only communicate with the one distant location on a leased line, but he or she can communicate with many devices when he or she has access to a switched line.

Acoustic Couplers

Although popular with data terminal and personal computer users in the early 1980s, today only a small percentage of transmissions use acoustic couplers. The *acoustic coupler* is a modem whose connection to the telephone line is obtained by acoustically coupling the telephone headset to the coupler. The primary advantage of the acoustic coupler is that it requires no hard-wired connection to the switched telephone network, enabling terminals and personal computers to be portable for their data transmissions. Due to the growth in modular telephone jacks, modems that interface the switched telephone network via a plug/jack arrangement in effect are portable devices. Because many hotels and older office buildings still have hard-wired telephones, the acoustic coupler enables terminal and personal computer users to communicate regardless of the method used to connect a telephone set to the telephone network.

The acoustic coupler converts the signals generated by a personal computer into a series of audible tones, which are then passed to the mouthpiece or transmitter of the telephone and in turn onto the switched telephone network. Information transmitted from the device at the other end of the data link is converted into audible tones at the earpiece of the telephone connected to the acoustic coupler. The coupler then converts those tones into the appropriate electrical signals recognized by the attached computer. The interrelationship of personal computers, acoustic couplers, modems, and analog transmission media is illustrated in Figure 13.5. Note the circle with the X in it in this figure. Here and in the remainder of this book, this symbol denotes the PSTN or switched telephone network.

Analog Facilities

Several types of analog switched facilities are offered by communications carriers. Each type of facility has its own set of characteristics and rate structures. Normally, for

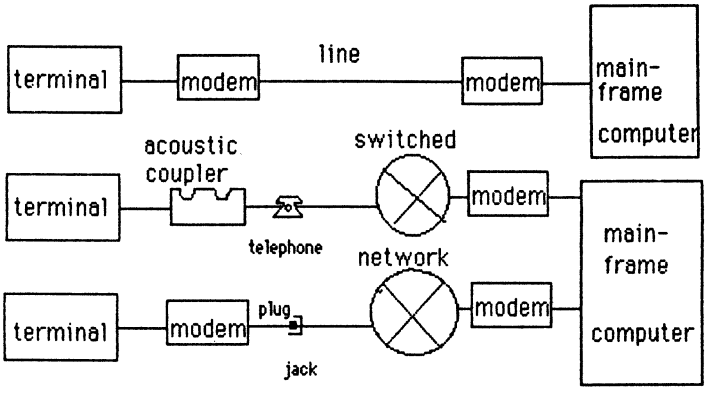


Figure 13.5
Interrelationship of Personal Computers, Terminals, Modems, Acoustic Couplers, Computers, and Analog Transmission Mediums

extensive communications requirements, companies conduct an analytic study to determine which type or types of service provide the optimal cost-effective service. The common types of analog switched facilities are direct distance dialing, wide area telephone service, and foreign exchange service.

With direct distance dialing (DDD) you can dial directly any telephone connected to the public switched telephone network. The dialed telephone may be connected to another terminal device or mainframe computer. The charge for this service, in addition to installation costs, may be a fixed monthly fee if no long distance calls are made. The fee usually is a message unit rate based on the number and duration of local calls, or a fixed fee plus any long distance charges incurred. Depending on the time of day a long distance call is initiated and its destination (intrastate or interstate), discounts from normal long distance tolls are available for selected calls made without operator assistance.

WATS

Introduced by AT&T for interstate use in 1961, Wide Area Telephone Service (WATS) is now offered by several long distance communications carriers. Its scope of coverage has been extended from the continental United States to Hawaii, Alaska, Puerto Rico, the U.S. Virgin Islands, and Europe, as well as selected Pacific and Asian countries.

Types of WATS

WATS is available in two forms, each designed for a particular type of communications requirement. Outward WATS is used when a specific location requires placing a large number of outgoing calls to geographically distributed locations. Inward WATS service provides the reverse capability, permitting a number of geographically distributed locations to communicate with a common facility. Calls on WATS lines are initiated in the same manner as a call placed on the public switched telephone network. However, instead of being charged on an individual call basis, the user of WATS facilities pays a flat rate per hour based on the number of communications hours per month occurring during weekday, evening, and night and weekend time periods.

A voice-band line called an *access line* is provided to the WATS user. This line links the user to a telephone company central office. Other than cost considerations and certain geographical calling restrictions that are a function of the service area of the WATS line, you can place as many calls as you like on this access line if the service is outward WATS, or you can receive as many calls as desired if the service is inward. Inward WATS, the well-known "800 toll-free number," enables remote callers to dial your facility toll free from the service area provided by the particular inward WATS-type of service selected. The charge for WATS is a function of the service area. This can be *intrastate WATS*, a group of states bordering the state where your main facility is located, a grouping of distant states, or *international WATS*, which extends inbound 800 service to the United States from selected overseas locations.

Another service very similar to WATS is AT&T's 800 READYLINE service. This service is essentially similar to WATS; however, calls can originate from or be directed to an existing telephone in place of the access line required for WATS service. Figure 13.6 illustrates the AT&T WATS *service area one* for the state of Georgia. If this service area is selected and a user in Georgia requires inward WATS service, he or she pays for toll free calls originating in the states surrounding Georgia—Florida, Alabama, Mississippi, Tennessee, Kentucky, South Carolina, and North Carolina. Similarly, if outward WATS service is selected for service area one, a person in Georgia connected to the WATS access line can use it to dial all telephones in the states previously mentioned. The states composing a service area vary, based on the state in which the WATS access line is installed. Thus, the states in service area one when an access line is in New York obviously differ from the states in a WATS service area one for an access line in Georgia. Fortunately, AT&T publishes a comprehensive book that includes 50 maps of the United States, illustrating the composition of the service areas for each state. Similarly, a time-of-day rate schedule for each state based on state service areas is also published by AT&T.

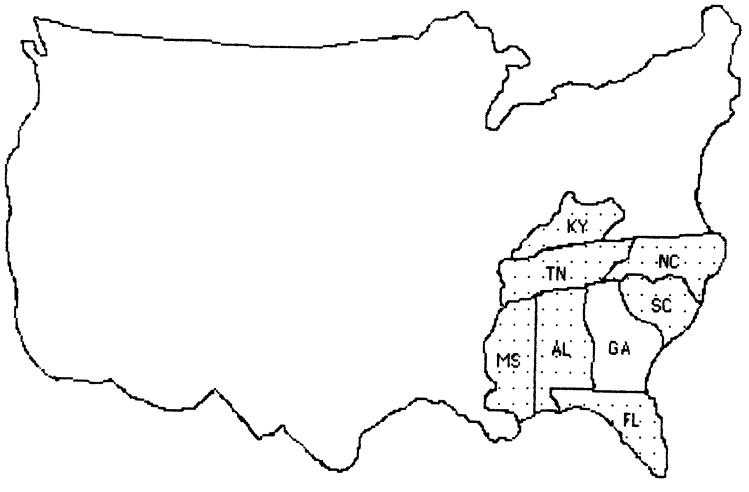


Figure 13.6
AT&T WATS Service Area One Access Line Located in Georgia

Applications of WATS for Computer Systems

In general, because WATS is a service based on volume usage, its cost per hour is less than the cost associated with the use of the PSTN for long distance calls. Thus, one common application for the use of WATS facilities is to install one or more inward WATS access lines at a mainframe location and have terminal and personal computer users distributed over a wide geographical area use the inward WATS facilities to access the mainframe computer.

Because international 800 service enables employees and customers of U.S. companies to call them toll free from foreign locations, this service may experience a considerable amount of data communications usage. This usage normally includes applications requiring access to such databases as hotel and travel reservation information, order entry and catalog sales data updating by persons traveling overseas with portable personal computers, and office personnel using terminals and personal computers in foreign countries who desire to access computational facilities and information utilities in the United States.

FX

Foreign exchange (FX) service provides a method of transmission from a group of terminal devices remotely located from a central computer facility, usually at less than the cost of direct distance dialing. An FX line can be viewed as a mixture of an analog switched and a leased line. To use an FX line, you dial a local number that is answered if the FX line is not in use. From the FX, the information is transmitted via a dedicated voice line to a permanent connection in the switching office of a communications carrier near the facility with which communication is desired. A line from the local switching office that terminates at the user's home office is included in the basic FX service. This is illustrated in Figure 13.7.

The use of an FX line eliminates long distance charges of direct dialing the distant computer facility. The major difference between an FX line and a leased line is that

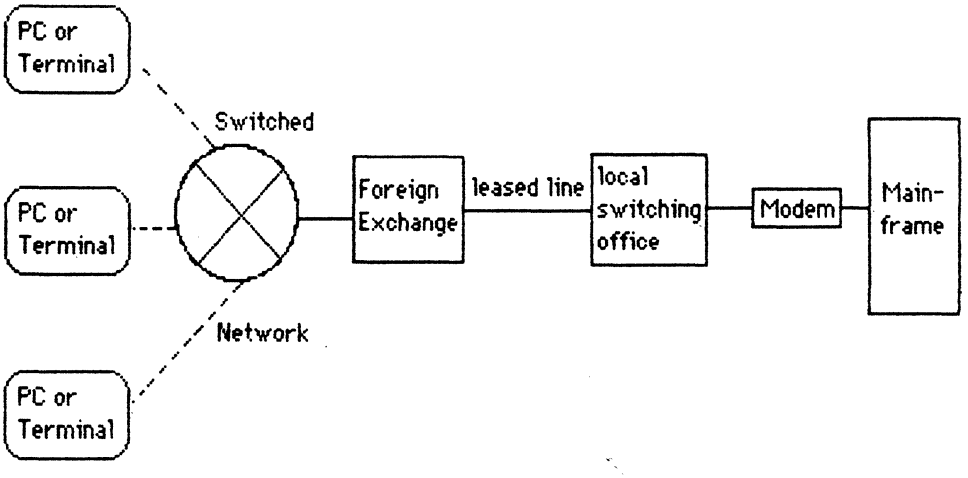


Figure 13.7
Foreign Exchange (FX) Service

any personal computer or terminal device dialing the FX line provides the second modem required for the transmission of data over the line, whereas a leased line used for data transmission normally has a fixed modem attached at both ends of the circuit.

Digital Facilities

In addition to analog service, numerous digital service offerings have been implemented by communications carriers over the last decade. Using digital services, data is transmitted from source to destination in digital form without converting the signal into an analog form for transmission over analog facilities (as when modems or acoustic couplers are interfaced to analog facilities).

In the United States, AT&T offers several digital transmission facilities under the ACCUNET Digital Service offering. Dataphone Digital Service was the charter member of the ACCUNET family and is deployed in 103 major metropolitan areas in the United States, as well as having an interconnection to Canada's digital network. Dataphone Digital Service operates at synchronous data transfer rates of 2.4, 4.8, 9.6, and 56 kilobits per second (kbps), providing users of this service with dedicated, two-way simultaneous transmission capability.

Dataphone Digital Service users can terminate their digital facilities with either a digital service unit or a channel service unit. A digital service unit (DSU) provides a standard interface to a digital transmission service and handles such functions as signal translation, regeneration, reformatting, and timing. Like the Dataphone Digital Service, the DSU is designed to operate at 2.4, 4.8, 9.6, and 56 kbps. The transmitting portion of the DSU processes the customer's signal into bipolar pulses suitable for transmission over the digital facility. The receiving portion of the DSU is used both to extract timing information and to regenerate mark and space information from the received bipolar signal.

The second interface arrangement for AT&T's Dataphone Digital Service is called a *channel service unit (CSU)* and is provided by the communication carrier to those customers who wish to perform the signal processing to and from the bipolar line, as well as to retime and regenerate the incoming line signals through the utilization of their own equipment.

As data is transmitted over digital facilities, the signal is regenerated by the communications carrier numerous times prior to its arrival at its destination. In general, digital service gives data communications users improved performance and reliability compared with analog service, due to the nature of digital transmission and the design of digital networks. This improved performance and reliability is because digital signals are regenerated, whereas when analog signals are amplified, any distortion to the analog signal is also amplified.

Although digital service is offered in many locations, for those locations outside the serving area of a digital facility you have to use analog devices as an extension to interface to the digital facility. The use of digital service via an analog extension is illustrated in Figure 13.8. The figure shows that if the closest city to the terminal located in city 2 that offers digital service is city 1, to use digital service to communicate with the computer an analog extension must be installed between the terminal location and city 1. In such cases, the performance, reliability, and possible cost advantages of

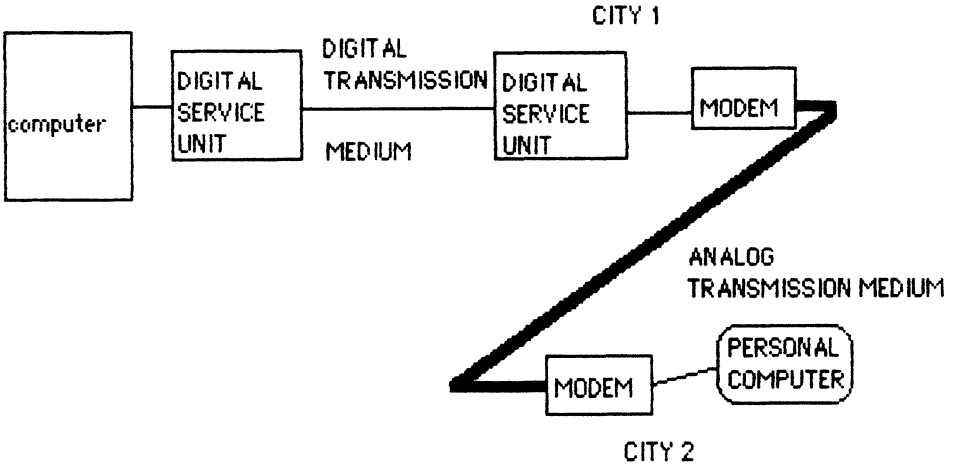


Figure 13.8
Analog Extension to Digital Service

using digital service may be completely eliminated. Digital service is available for both switched network and leased service. A leased digital line is similar to a leased analog line in that it is dedicated for full-time use to a particular user.

Transmission Mode

One method of characterizing lines, terminal devices, mainframe computers, and modems is by their transmission or communications mode. The three classes of transmission modes are simplex, half-duplex, and full-duplex.

Simplex transmission transfers data in one direction only, so the receiver of information cannot respond to the transmission. A home AM radio that receives a signal transmitted from a radio station is an example of a simplex communications mode. In a data transmission environment, simplex transmission might be used to turn on or off specific devices at a certain time or when a certain event occurs. For example, a computer-controlled environmental system may feature a furnace turned on or off depending on the thermostat setting and the current temperature in various parts of a building. Normally, simplex transmission is not used where human-machine interaction is required, because this transmitter cannot enable the receiver to reply to the originator.

Half-duplex transmission permits transmission in either direction; however, transmission can occur in only one direction at a time. Half-duplex transmission is used in citizens band (CB) radio transmission, in which the operator can either transmit or receive but cannot perform both functions at the same time on the same channel. When the operator has completed a transmission, the other party must be advised that he or she is through transmitting and is ready to receive by saying *over*. Then the other operator can begin transmission.

When data is transmitted over the telephone network, the transmitter and the receiver of the modem or acoustic coupler must be appropriately turned on and off as the direction of the transmission varies. Both simplex and half-duplex transmission require two wires to complete an electrical circuit. The upper diagram in Figure 13.9 illustrates a half-

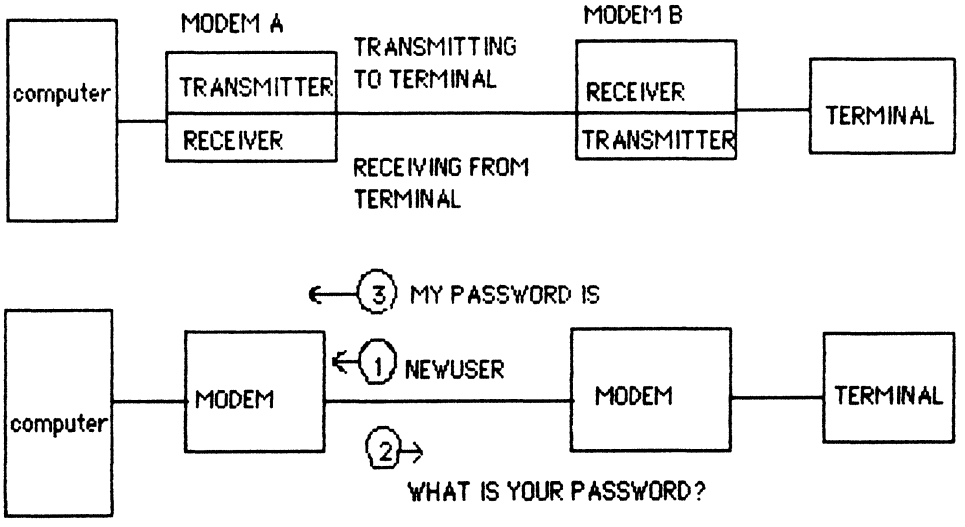


Figure 13.9 Half-Duplex Transmission

duplex modem interconnection, whereas the lower portion of that illustration shows a typical sequence of events that might occur during the sign-on process to access a mainframe computer. In the sign-on process, you might first transmit the word NEWUSER to inform the mainframe computer that a new user wishes to establish a connection. The computer responds by asking for the user's password, which is then furnished. In the top portion of Figure 13.9, when data is transmitted from a mainframe computer to a terminal or personal computer, control signals are sent from the mainframe computer to modem A, which turns on the modem A transmitter and causes the modem B receiver to respond.

When data is transmitted from the personal computer or terminal to the mainframe computer, the modem B receiver is disabled and its transmitter is turned on. The modem A transmitter is disabled and its receiver becomes active. The time necessary to effect these changes is called *transmission turnaround time,* and during this interval transmission is temporarily halted. Half-duplex transmission can occur on either a 2-wire or 4-wire circuit. The switched (PSTN) network is a 2-wire circuit, whereas leased lines can be either 2-wire or 4-wire links. A 4-wire circuit is essentially a pair of 2-wire links that can be used for transmission in both directions simultaneously. This type of transmission occurs in *full-duplex mode.*

Half-duplex communications can occur on either a 2-wire or 4-wire circuit. Although one would normally expect full-duplex transmission to be accomplished over a 4-wire connection that provides two 2-wire paths, full-duplex transmission can also occur on a 2-wire connection. This is accomplished by the use of modems that subdivide the frequency bandwidth of the 2-wire connection into two distinct channels, permitting simultaneous data flow in both directions on a 2-wire circuit.

Most modems that operate at data rates up to 2400 bps and are designed for use on the PSTN subdivide the 3000 Hz (Hertz) bandwidth of a telephone channel into two subchannels by frequency. On one subchannel the binary marks (1's) and spaces (0's) transmitted by a digital service are modulated by a modem into two distinct frequency

tones—Tm and Ts. The modem at the opposite end of the line is designed to receive the frequencies Rm and Rs that correspond to the frequencies the first modem transmits its marks and spaces on. Similarly, the second modem transmits the marks and spaces of a digital device connected to it at two higher frequencies—Tm1 and Ts1. The first modem's receiver is designed so its receiver recognizes tones at those frequencies, receiving data at Rm1 and Rs1 that correspond to the frequencies the second modem transmits.

To distinguish frequency settings, modems that subdivide the bandwidth in this manner are categorized as originate mode and answer mode. Each modem category denotes the transmit and receive frequencies the modem is built to operate with. By convention, a personal computer or terminal uses an *originate mode modem,* whereas a mainframe computer uses an *answer mode modem.* To permit PCs to talk with one another, many modems are dual-mode devices. That is, they can be set to originate mode or answer mode operations. Then, one modem must be in its originate mode of operation, whereas the other modem must be placed in its answer mode of operation to obtain frequency compatibility between the modems.

Full-duplex transmission is often used when large amounts of alternate traffic must be transmitted and received within a fixed time period. If two channels were used in the CB example—one for transmission and another for reception—two simultaneous transmissions could be effected. Whereas full-duplex transmission provides more efficient throughput, this efficiency may be negated by the cost of software to obtain this capability and more complex equipment required by this mode of transmission. In Figure 13.10, the three types of transmission modes are illustrated, and Table 13.3 summarizes these modes.

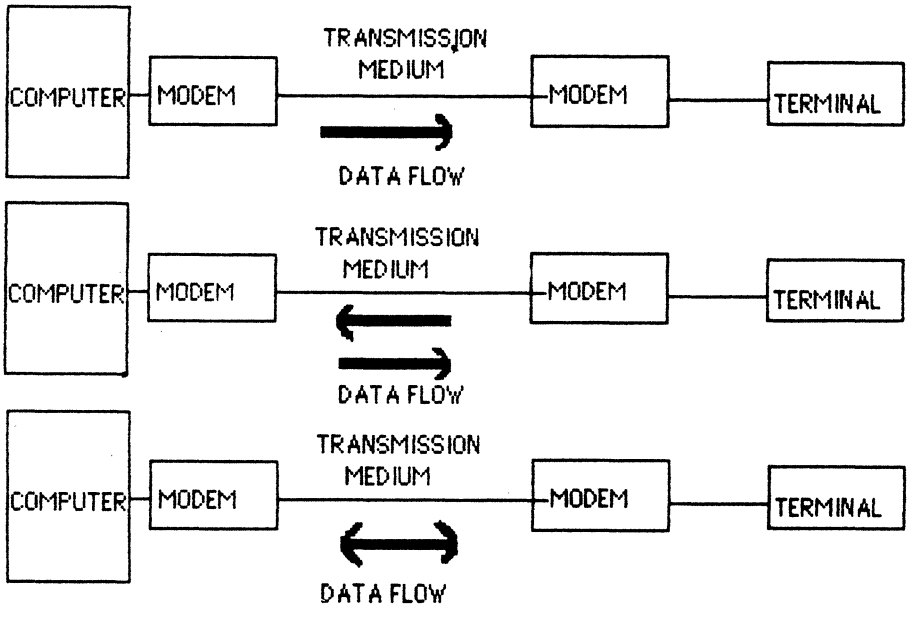


Figure 13.10
Transmission Modes

Table 13.3
Transmission Mode Comparison

Symbol	ANSI	US Telecommunications Industry	CCITT	Historical Physical Line Requirement
←	One-way only	Simplex		2-wire
⇆	Two-way alternate	Half-duplex (HDX)	Simplex	2-wire
↔	Two-way simultaneous	Full-duplex (FDX)	Duplex	4-wire

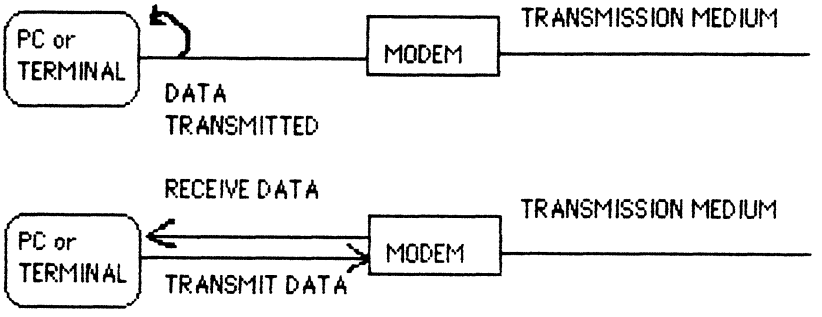
Note that the column CCITT in Table 13.3 refers to Consultive Committee on International Telephone and Telegraph. The CCITT operates as part of the International Telecommunications Union (ITU), which is a United Nations agency. Because CCITT modem standards are primarily followed in Europe, these standards may be of particular interest to people or companies in the United States that communicate with overseas locations or ship equipment purchased in the United States to overseas destinations.

PC and Terminal Operations with Mainframe Computers

To refer solely to personal computer and terminal operations, the terms *half-duplex* and *full-duplex* operation take on meanings different from the communications mode of the transmission medium. Vendors commonly use half-duplex to denote that the computer or terminal device is in a local copy mode of operation. This means that each time a key is pressed on the keyboard its character is printed or displayed on the local personal computer or terminal, as well as transmitted. Thus, a terminal device operated in a half-duplex mode displays each character printed or displayed on its monitor as it is transmitted.

When a personal computer or terminal is in full-duplex mode, each character typed is transmitted but not immediately displayed or printed. Here the device on the distant end of the transmission path must "echo" the character back to the originator, which on receipt displays or prints the character. Thus, a personal computer or terminal in a full-duplex mode of operation would only print or display the characters typed on a keyboard after they are echoed back by the device at the other end of the line. Figure 13.11 illustrates full- and half-duplex devices as they apply to terminals. Note that

Figure 13.11
Terminal Operation Modes



although most conventional terminals have a switch to control the duplex setting of the device, personal computer users normally obtain their duplex setting via the software they are using.

Half- and full-duplex in terms of mainframe computer systems normally refer to whether the device echoes received characters back to the originator. A half-duplex computer system does not echo characters, whereas a full-duplex computer system echoes each character it receives.

When you consider the operating mode of the terminal device, the transmission medium, and the operating mode of the mainframe computer receiving the transmission, three things could occur in response to each character you type. Assuming a transmission medium is employed that can be used for either half- or full-duplex communications, a terminal device could print or display no character for each character transmitted, one character for each character transmitted, or two characters for each character transmitted. Here the resulting character(s) printed or displayed would be dependent on the operating mode of the terminal device and the host computer connected to, as indicated in Table 13.4.

To understand the character display column in Table 13.4, examine the two-character display that results when the terminal device operates in a half-duplex mode and the mainframe computer operates in a full-duplex mode.

When a personal computer or terminal is in a half-duplex mode, it echoes each transmitted character onto its printer or display. At the other end of the communications path, if the mainframe computer is in a full-duplex mode of operation it echoes the received character to the PS/2 or terminal, causing a second copy of the transmitted character to be printed or displayed. Thus, two characters are output to the printer or screen for each character transmitted. To alleviate this situation, you can change your terminal's transmission mode to full-duplex. Normally you do this by turning "echo" off during the initialization of your communications program if you are using a personal computer; or you turn a switch to half-duplex if you are operating a conventional terminal.

Transmission Techniques

Data can be transmitted either synchronously or asynchronously. *Asynchronous transmission* is commonly referred to as a *start-stop transmission*, in which one character at a time is transmitted or received. Start and stop bits are used to separate characters

Table 13.4
Operating Mode and Character Display

Terminal Device	Operating Mode, Host Computer	Character Display
Half-duplex	Half-duplex	1 character
Half-duplex	Full-duplex	2 characters
Full-duplex	Half-duplex	No characters
Full-duplex	Full-duplex	1 character

and synchronize the receiver with the transmitter, thus helping to reduce the possibility of data becoming garbled. Most teletype compatible devices designed for human–machine interaction transmit data asynchronously. Teletype compatibility means that your PS/2 operates similar to (or *emulates*) the Teletype terminal manufactured by Western Electric, a subsidiary of AT&T. Various versions of this popular terminal have been manufactured for over 30 years, and an installed base of approximately one million such terminals is in operation worldwide. As characters are typed on the device's keyboard they are transmitted to the computer, with idle time occurring between the transmission of characters. This is illustrated in the bottom of Figure 13.12.

Asynchronous Transmission

In asynchronous transmission, each character to be transmitted is encoded into a series of pulses. The transmission of the character is started by a start pulse equal in length to a code pulse. The encoded character (series of pulses) is followed by a stop pulse that may be equal to or longer than the code pulse, depending on the transmission code used.

The start bit represents a transition from a mark to a space. Because in an idle condition when no data is transmitted the line is held in a marking condition, the start bit serves as an indicator to the receiving device that a character of data follows. Similarly, the stop bit causes the line to be placed back into its previous "marking" condition, signifying to the receiver that the data character is completed.

As illustrated in the top portion of Figure 13.12, the transmission of an 8-bit character requires either 10 or 11 bits, depending on the length of the stop bit. In the start-stop mode of transmission, transmission starts anew for each character and stops after each character. This is indicated in the lower portion of Figure 13.12. Because synchronization starts anew with each character, any timing discrepancy is cleared at the end of each

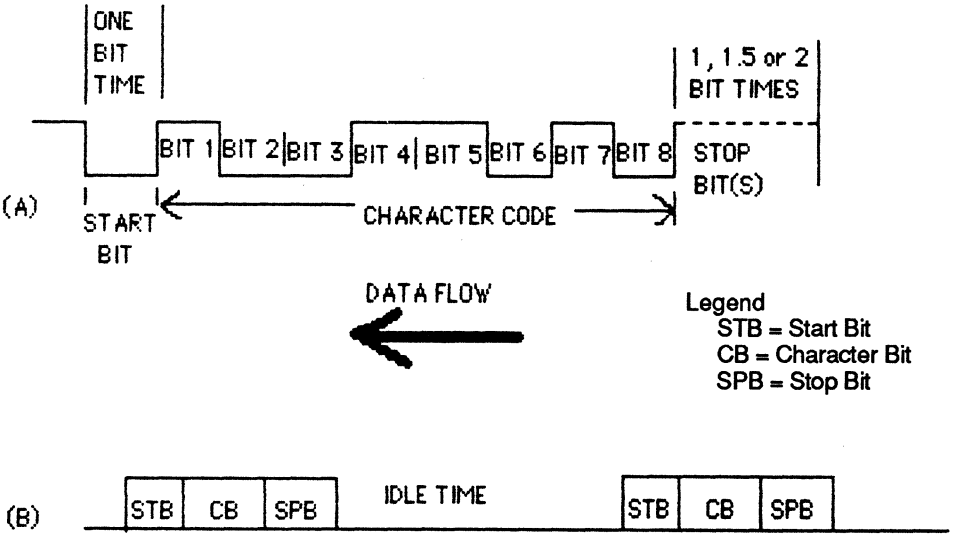


Figure 13.12
Asynchronous (Start-Stop) Transmission

character, and synchronization is maintained on a character-by-character basis. Asynchronous transmission normally is used for transmission at speeds up to 9600 bps over the switched telephone network, whereas data rates up to 19,200 bps are possible over a direct connect cable whose distance is limited to 50 feet. Such transmissions can also occur on conditioned leased lines.

The terms *asynchronous TTY* and *TTY compatible* refer to the asynchronous start-stop protocol employed originally by Teletype terminals. It is the protocol in which data is transmitted on a line-by-line basis between a terminal device and a mainframe computer. In comparison, more modern terminals with cathode ray tube (CRT) displays are usually designed to transfer data on a full screen basis.

Personal computer users only require an asynchronous communications adapter and a software program that transmits and receives data on a line-by-line basis to connect to a mainframe that supports asynchronous TTY compatible terminals. Here the software program that transmits and receives data on a line-by-line basis is normally referred to as a *TTY emulator program* and is the most common type of communications program written for use with personal computers.

Synchronous Transmission

A second type of transmission involves sending a grouping of characters in a continuous bit stream. This type of transmission is referred to as *synchronous* or *bit-stream synchronization.* In the synchronous mode of transmission, modems located at each end of the transmission medium normally provide a timing signal or clock to establish the data transmission rate and enable the devices attached to the modems to identify the appropriate characters as they are being transmitted or received. In some instances, timing may be provided by the terminal device itself or a communication component, such as a multiplexer or front-end processor channel. No matter what timing source is used, prior to beginning the transmission of data the transmitting and receiving devices must establish synchronization among themselves. To keep the receiving clock in step with the transmitting clock for the duration of a stream of bits representing a large number of consecutive characters, the transmission of the data is preceded by the transmission of one or more special characters. These special synchronization or SYN characters (described fully later in the chapter) are at the same code level (number of bits per character) as the coded information to be transmitted. However, they have a unique configuration of zero and one bits that are interpreted as the SYN character. Once a group of SYN characters is transmitted, the receiver recognizes and synchronizes itself onto a stream of those SYN characters.

After synchronization is achieved, actual data transmission can proceed. Synchronous transmission is illustrated in Figure 13.13. In synchronous transmission, characters are

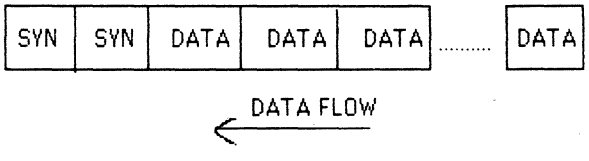


Figure 13.13
Synchronous Transmission

grouped or blocked into groups of characters, requiring a buffer or memory area so characters can be grouped together. In addition to having a buffer area, more complex circuitry is required for synchronous transmission, because the receiving device must remain in phase with the transmitter for the duration of the transmitted block of information. Synchronous transmission is normally used for data transmission rates in excess of 2000 bps. The major characteristics of asynchronous and synchronous transmission are denoted in Table 13.5.

Types of Transmission

The two types of data transmission one can consider are serial and parallel. For *serial transmissions* the bits that compose a character are transmitted in sequence over one line, whereas in *parallel transmissions* characters are transmitted serially but the bits that represent the character are transmitted in parallel. If a character consists of eight bits, parallel transmission requires a minimum of eight lines. Additional lines may be necessary for control signals or for the transmission of a parity bit. Although parallel transmission is used extensively in computer-to-peripheral unit transmission, it is not normally employed other than in dedicated data transmission usage, due to the cost of the extra circuits required.

A typical use of parallel transmission is the in-plant connection of badge readers and similar devices to a computer in that facility. Parallel transmission may also reduce the cost of terminal circuitry, because the terminal does not have to convert the internal character representation to a serial data stream for transmission. However, the cost of the transmission medium and interface increases due to the additional number of conductors required. Because the total character can be transmitted at the same moment in time using parallel transmission, you can obtain higher data transfer rates than are

Table 13.5
Transmission Technique Characteristics

Asynchronous
1. Each character is prefixed by a start bit and followed by one or more stop bits.
2. Idle time (period of inactivity) can exist between transmitted characters.
3. Bits within a character are transmitted at prescribed time intervals.
4. Timing is established independently in the computer and terminal.
5. Transmission speeds normally do not exceed 9600 bps over switched facilities and 19,200 bps over dedicated links and leased lines.

Synchronous
1. SYN characters prefix transmitted data.
2. SYN characters are transmitted between blocks of data to maintain line synchronization.
3. No gaps exist between characters.
4. Timing is established and maintained by the transmitting and receiving modems, the terminal, or other devices.
5. Terminals must have buffers.
6. Transmission speeds normally are in excess of 2000 bps.

possible with serial transmission facilities. For this reason, most local facility communications between computers and their peripheral devices are accomplished using parallel transmission. In comparison, communications between personal computers and other computers normally occurs serially, because this requires only one line to interconnect two devices for communications. Figure 13.14 illustrates serial and parallel transmission.

Line Structure

The geographical distribution of personal computers and terminal devices and the distance between each device and the device it transmits to are important parameters to consider in developing a network configuration. The method used to interconnect personal computers and terminals to mainframe computers or to other devices is known as *line structure* and is the basis of a computer's network configuration. The two types of line structure used in networks are *point-to-point* and *multipoint*, the latter also known as *multidrop*. Communications lines that only connect two points are point-to-point lines. An example of this line structure is depicted at the top of Figure 13.15. As illustrated, each personal computer or terminal transmits and receives data to and from a computer via an individual connection that links a specific PC or terminal to the computer. The point-to-point connection can use a dedicated circuit or a leased line, or it can be obtained via a connection initiated over the switched (dial-up) telephone network.

When two or more terminal locations share portions of a common line, the line is a multipoint or multidrop line. Although no two devices on such a line can transmit data at the same time, two or more devices may receive a message at the same time. The number of devices receiving such a message is dependent on the addresses assigned to

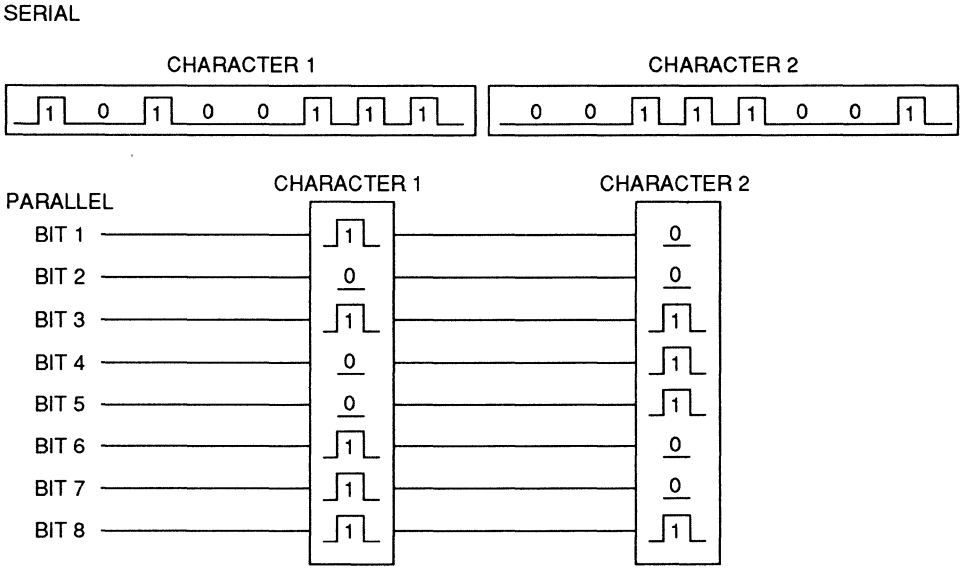


Figure 13.14 Types of Data Transmission

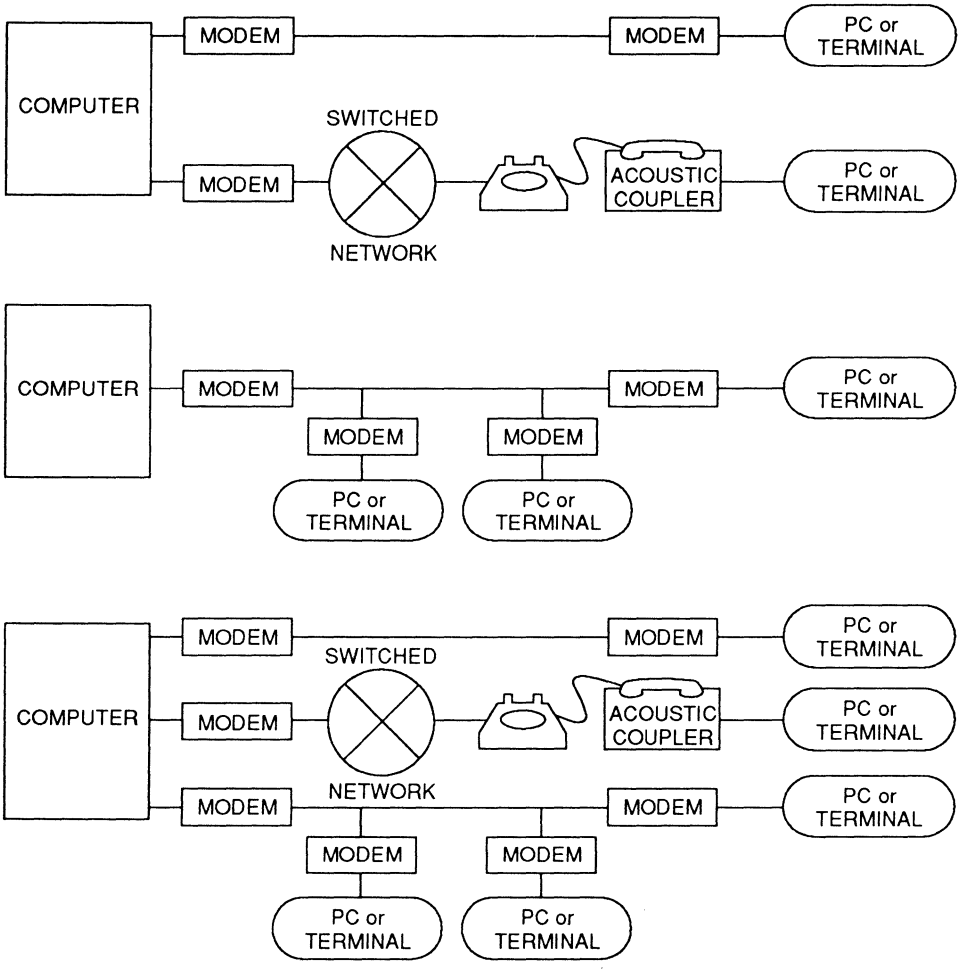


Figure 13.15
Line Structures in Networks

the message recipients. In some systems a "broadcast" address permits all devices connected to the same multidrop line to receive a message at the same time. When you use multidrop lines, overall line costs may be reduced, because common portions of the line are shared for use by all devices connected to that line. To prevent data transmitted from one device from interfering with data transmitted from another device, a *line discipline* or control must be established for such a link.

This discipline controls transmission so no two devices transmit data at the same time. A multidrop line structure is depicted in the center diagram of Figure 13.15. For a multidrop line that links n devices to a mainframe computer, $n + 1$ modems are required, one for each device and one located at the computer facility.

Both point-to-point and multipoint lines may be intermixed in developing a network, and transmission can be either in the full- or half-duplex mode. This mixed line structure is shown in the lower portion of Figure 13.15.

Line Discipline

When several devices share the use of a common, multipoint communications line, only one device may transmit at any one time, although multiple devices may receive information simultaneously. To prevent two or more devices from transmitting at the same time, a technique known as *poll and select* is employed as the line discipline for multidrop lines. To use poll and select, each device on the line must have a unique address of one or more characters, as well as circuitry to recognize a message sent from the computer to that address. When the computer polls a line, in effect it asks each device in a predefined sequence if it has data to transmit. If the device has no data to transmit, it informs the computer and the computer continues its polling sequence until it encounters a device on the line that has data to send. Then the computer acts on that data transfer.

As the computer polls each device, the other devices on the line must wait until they are polled before they can be serviced. Conversely, to transmit data from the computer to each device on a multidrop line, the computer selects the device address to which that data is to be transferred, informing the device that data is to be transferred to it, then transmitting data to the selected device. You can use polling and selecting to service both asynchronous or synchronous operating terminal devices connected to independent multidrop lines. Due to the control overhead of polling and selecting, synchronous high-speed devices are normally serviced in this type of environment. Using signals and procedures, polling and selecting line control ensures the orderly and efficient use of multidrop lines. An example of a computer polling the second personal computer or terminal on a multipoint line and receiving data from that device is shown in the upper diagram of Figure 13.16. The lower portion of that illustration shows the computer first selecting the third device on the line and then transferring a block of data to that device.

When personal computers and terminals transmit data on a point-to-point line to a computer or another terminal, the transmission of that data occurs at the discretion of the terminal operator. This method of line control is known as *nonpoll and select* or *free-wheeling* transmission.

Transmission Rate

Many factors can affect the transmission rate at which data is transferred. The types of modem or acoustic coupler used as well as the line discipline and the type of communications adapter installed in one's personal computer all play governing roles that affect transmission rates; however, the transmission medium itself is an important factor in determining transmission rates.

Data transmission services offered by communications carriers such as AT&T and MCI are based on their available plant facilities. Depending on personal computer and mainframe computer locations, two types of transmission services may be available. Analog transmission is most readily available and can be employed on switched or leased telephone lines. Digital transmission is only available in most large cities, and analog extensions are required to connect to this service from nondigital service locations, as

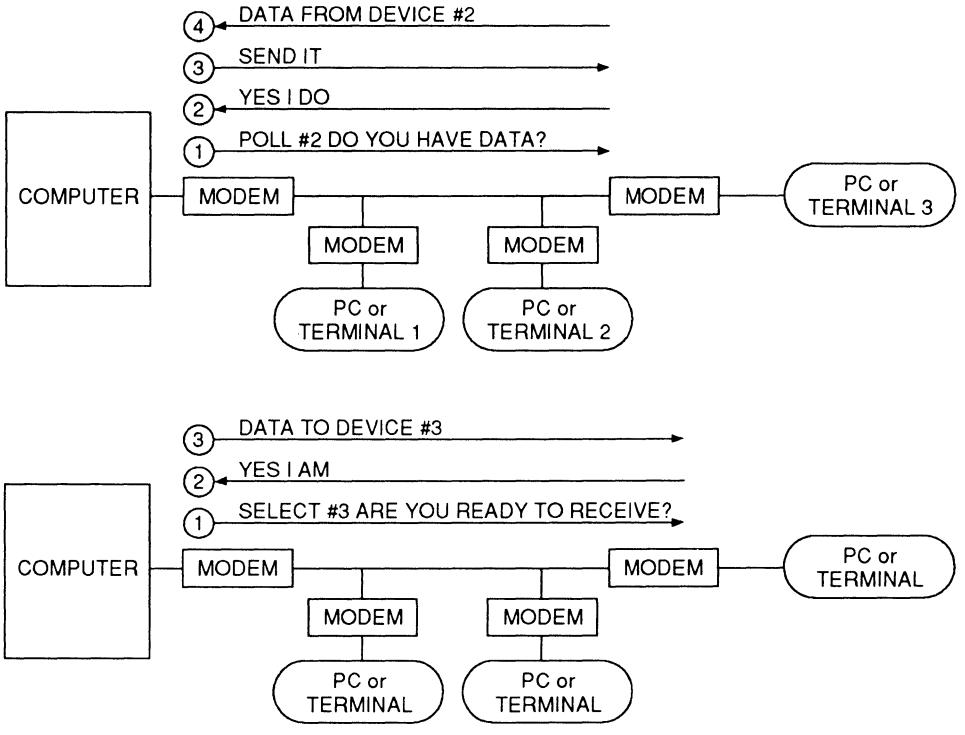


Figure 13.16
Poll and Select Line Discipline

previously illustrated in Figure 13.8. Within each type of service several grades of transmission are available for consideration.

In general, analog service offers the user three grades of transmission: narrowband, voice band, and wideband. The data transmission rates using each of these grades of service are dependent on the bandwidth and electrical properties of each type of circuit offered within each grade of service. Basically, transmission speed is a function of the bandwidth of the communications line: the greater the bandwidth, the higher the possible speed of transmission.

Narrowband facilities are obtained by the carrier subdividing a voice band circuit or by grouping a number of transmissions from different users onto a single portion of a circuit by time. Transmission rates obtained on narrowband facilities range between 45 and 300 bps. Teletype terminals that connect to message switching systems are a primary example of narrowband facilities.

Whereas narrowband facilities have a bandwidth in the range of 200 to 400 Hz, voice band facilities have a bandwidth in the range of 3000 Hz. Data transmission speeds obtainable on voice band facilities are differentiated by the type of voice band facility used—switched dial-up transmission or transmission via a leased line. For transmission over the switched telephone network, maximum data transmission is normally under or up to 9600 bps, with data rates up to 100 percent higher obtainable when transmission occurs through modern electronic switches instead of the older, electromechanical switches still used in many telephone offices. Because leased lines can be

conditioned, a speed of 9600 to 19,200 bps is frequently obtainable on such lines. Although low data speeds can be transmitted on both narrowband and voice band circuits, do not confuse the two—a low data speed on a voice circuit is transmission at a rate far less than the maximum permitted by that type of circuit. On the other hand, a low rate on a narrowband facility is at or near the maximum transmission rate permitted by that type of circuit.

Facilities that have a higher bandwidth than voice band are termed *wideband* or *group-band facilities,* because they provide a wider bandwidth through the grouping of a number of voice band circuits. Wideband facilities are available only on leased lines and permit transmission rates in excess of 19,200 bps. Transmission rates on wideband facilities vary with the offerings of communications carriers. Speeds normally available include 19.2, 40.8, 50, 230.4 kbps, and 1.544 Mbps. Due to the data rates on such lines normally exceeding the serial data rate obtainable with a personal computer communications adapter, PCs are very rarely connected to wideband facilities.

For direct connect circuits, transmission rates are a function of the distance between the personal computer and the mainframe computer, as well as the gauge of the conductor used and the type of communications adapter installed in the PC.

In the area of digital service, several offerings are currently available for user consideration. Digital data service is offered by AT&T as Dataphone Digital Service (DDS). It provides interstate, full-duplex, point-to-point, and multipoint leased line as well as synchronous digital transmission at speeds of 2400, 4800, 9600, and 56,000 bps. It also provides data transmission at 1.544 million bps between the servicing areas of many digital cities.

A new high-speed AT&T digital switched communications service offers full-duplex, synchronous transmission over a common digital network at a transmission rate of 56,000 bps. Already, two companies are marketing adapter boards designed to enable IBM and compatible personal computers to communicate over this digital facility. When equipped with such adapter boards, PCs can be used as terminals for high speed data communications, desktop video conferencing, or as a gateway to a local area network. Table 13.6 lists the main analog and digital facilities, the range of transmission speeds over those facilities, and the general use of such facilities.

Transmission Codes

Data within a computer is structured according to the architecture of the computer. The internal representation of data in a computer is seldom suitable for transmission to devices other than the peripheral units attached to the computer. In most cases, to effect data transmission, internal computer data must be redesigned or translated into a suitable transmission code. This transmission code creates a correspondence between the bit encoding of data for transmission or internal device representation and printed symbols. The result of the translation is usually dictated by the character code used by the remote device the personal computer is attempting to access. Frequently available codes include Baudot, which is a 5-level (5 bits per character) code; binary-coded decimal (BCD), which is a 6-level code; American Standard Code for Information Interchange (ASCII), which is normally a 7-level code; and the Extended Binary-Coded Decimal Interchange code (EBCDIC), which is an 8-level code.

Table 13.6 Common Transmission Facilities

Facility	Transmission Speed	Use
Analog		
Narrowband	45–300 bps	Message switching
Voice band		
Switched	Between 9600 and 19,200	Time sharing; information utility access; remote job entry
Leased	Up to 19,200 bps	File transfer operation
Wideband	Over 19,200 bps	Mainframe computer-to-computer; remote job entry; tape-to-tape transmission; high-speed terminal to high-speed terminal
Digital		
Leased line	2.4, 4.8, 9.6, 56 kbps and 1.544 Mbps	Remote job entry; mainframe computer-to-computer; high-speed facsimile
Switched	56 kbps	PC or terminal-to-PC or terminal; computer-to-computer; high-speed terminal to computer

In addition to information being encoded into a certain number of bits based on the transmission code used, the unique configuration of those bits to represent certain control characters can be considered as a code that can be used to effect line discipline. These control characters can indicate the acknowledgment of the receipt of a block of data without errors (ACK), the start of a message (SOH), or the end of a message (EXT), with the number of permissible control characters standardized according to the code employed. With the growth of personal computer to personal computer data transmission, a large amount of processing can be avoided by transferring the data in the format used by the computer for internal processing. Such transmission is known as *binary mode transmission, transparent data transfer, code-independent transmission,* or *native mode transmission.*

EBCDIC

The Extended Binary-Coded Decimal Interchange (EBCDIC) is an extension of an earlier BCD system and uses 8 bits for character representation. This code permits 2^8 or 256 unique characters to be represented, although currently a lesser number is assigned meanings. This code is primarily used for transmission by byte-oriented com-

puters, where a *byte* is a grouping of eight consecutive binary digits operated on as a unit by the computer. When computers use this code they may not need to convert the codes if personal computers or terminals accessing an EBCDIC computer operate with the same character set. Several subsets of EBCDIC exist that have been tailored for use with certain devices. As an example, IBM 3270 type terminal products do not use a paper feed, and its character representation is omitted in the EBCDIC character subset for that type of device, as indicated in Figure 13.17.

ASCII Code

As a result of the proliferation of data transmission codes, several attempts to develop standardized codes for data transmission have occurred. One such code is the American Standard Code for Information Interchange (ASCII). This 7-level code is based on a 7-bit code developed by the International Standards Organization (ISO) and permits 128 possible combinations or character assignments to include 96 graphic characters that are printable or displayable and 32 control characters to include device control and information transfer control characters.

Figure 13.18 lists the ASCII character set, while Table 13.7 lists the ASCII control characters by position and their meaning. A more detailed explanation of these control characters is contained in the section covering protocols in this chapter.

Figure 13.17
EBCDIC Code Implemented for the IBM 3270 Information Display System

Bits 4567 ↓ / Hex 1		00				01				10				11				← Bits 0,1
		00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11	← 2,3
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← Hex 0
0000	0	NUL	DLE			SP	&	-									0	
0001	1	SOH	SBA					/		a	j			A	J		1	
0010	2	STX	EUA		SYN					b	k	s		B	K	S	2	
0011	3	ETX	IC							c	l	t		C	L	T	3	
0100	4									d	m	u		D	M	U	4	
0101	5	PT	NL							e	n	v		E	N	V	5	
0110	6			ETB						f	o	w		F	O	W	6	
0111	7			ESC	EOT					g	p	x		G	P	X	7	
1000	8									h	q	y		H	Q	Y	8	
1001	9		EM							i	r	z		I	R	Z	9	
1010	A					¢	!	¦	:									
1011	B					.	$	,	#									
1100	C		DUP		RA	<	*	%	@									
1101	D		SF	ENQ	NAK	(	)	_	'									
1110	E		FM			+	;	>	=									
1111	F		ITB		SUB		¬	?	"									

Figure 13.18
The ASCII Character Set

b_7 b_6 b_5 / Bits b_4 b_3 b_2 b_1 / ROW	COLUMN 0	1	2	3	4	5	6	7
0 0 0 0 0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1 1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0 2	STX	DC2	"	2	B	R	b	r
0 0 1 1 3	ETX	DC3	#	3	C	S	c	s
0 1 0 0 4	EOT	DC4	$	4	D	T	d	t
0 1 0 1 5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0 6	ACK	SYN	&	6	F	V	f	v
0 1 1 1 7	BEL	ETB	'	7	G	W	g	w
1 0 0 0 8	BS	CAN	(	8	H	X	h	x
1 0 0 1 9	HT	EM	)	9	I	Y	i	y
1 0 1 0 10	LF	SUB	*	:	J	Z	j	z
1 0 1 1 11	VT	ESC	+	;	K	[	k	{
1 1 0 0 12	FF	FS	,	<	L	\	l	|
1 1 0 1 13	CR	GS	-	=	M	]	m	}
1 1 1 0 14	SO	RS	.	>	N	^	n	~
1 1 1 1 15	SI	US	/	?	O	_	o	DEL

Note that b_7 is the higher order bit and b_1 is the low order bit as indicated by the following example for coding the letter C.

b_7	b_6	b_5	b_4	b_3	b_2	b_1
1	0	0	0	0	1	1

Code Conversion

A frequent problem in data communications is that of code conversion. Consider what must be done to enable a mainframe computer with an EBCDIC character set to transmit and receive information from a terminal or personal computer with an ASCII character set. When the personal computer or terminal transmits a character, that character is encoded according to the ASCII character code. On receipt of that character, the mainframe computer must convert the bits of information of the ASCII character into an equivalent EBCDIC character. Conversely, when data is to be transmitted to the personal computer or terminal, it must be converted from EBCDIC to ASCII so the PC or terminal will be able to decode and act according to the information in the character that the terminal or personal computer is built to interpret.

Normally, ASCII to EBCDIC code conversion is implemented when an IBM PS/2 is required to operate as a 3270 type terminal. This type of terminal is typically connected to an IBM or IBM compatible mainframe computer and the terminal's replacement by an IBM PS/2 requires the PS/2's ASCII coded data to be translated into EBCDIC. There are many ways to obtain this conversion to include emulation boards that are

Table 13.7
ASCII Control Characters

Column/Row	Control Character	Mnemonic	Meaning[1]
0/0	^@	NUL	Null (CC)
0/1	^A	SOH	Start of Heading (CC)
0/2	^B	STX	Start of Text (CC)
0/3	^C	ETX	End of Text (CC)
0/4	^D	EOT	End of Transmission (CC)
0/5	^E	ENQ	Enquiry (CC)
0/6	^F	ACK	Acknowledge (CC)
0/7	^G	BEL	Bell
0/8	^H	BS	Backspace (FE)
0/9	^I	HT	Horizontal Tabulation (FE)
0/10	^J	LF	Line Feed (FE)
0/11	^K	VT	Vertical Tabulation (FE)
0/12	^L	FF	Form Feed (FE)
0/13	^M	CR	Carriage Return (FE)
0/14	^N	SO	Shift Out
0/15	^O	SI	Shift In
1/0	^P	DLE	Data Link Escape (CC)
1/1	^Q	DC1	Device Control 1
1/2	^R	DC2	Device Control 2
1/3	^S	DC3	Device Control 3
1/4	^T	DC4	Device Control 4
1/5	^U	NAK	Negative Acknowledge (CC)
1/6	^V	SYN	Synchronous Idle (CC)
1/7	^W	ETB	End of Transmission Block (CC)
1/8	^X	CAN	Cancel
1/9	^Y	EM	End of Medium
1/10	^Z	SUB	Substitute
1/11	^[	ESC	Escape
1/12	^/	FS	File Separator (IS)
1/13	^]	GS	Group Separator (IS)
1/14	^^	RS	Record Separator (IS)
1/15	^-	US	Unit Separator (IS)
7/15		DEL	Delete

1. (CC) Communications control; (FE) Format effecter; (IS) Information separator.

inserted into the system unit of the personal computer and protocol converters that are connected between the PC and the mainframe computer. In our examination of networking strategies later in this book, we will explore these and other methods that

enable the personal computer to communicate with mainframe computers that transmit data coded in EBCDIC.

Table 13.8 lists the ASCII and EBCDIC code character values for the 10 digits for comparison purposes. In examining the difference between ASCII and EBCDIC coded digits, note that each EBCDIC coded digit has a value precisely Hex C0 (decimal 192) higher than its ASCII equivalent. Although this might appear to make code conversion a simple process of adding or subtracting a fixed quantity depending on which way the code conversion takes place, in reality many of the same ASCII and EBCDIC coded characters differ by varying quantities. As an example, the slash (/) character is Hex 2F in ASCII and Hex 61 in EBCDIC, a difference of Hex 92 (decimal 146). In comparison, other characters, such as the carriage return and form feed, have the same coded value in ASCII and EBCDIC, and other characters are displaced by different amounts in these two codes. Due to this, code conversion is typically performed as a table lookup process, with two buffer areas used to convert between codes in each of the two conversion directions. Thus, one buffer area might have the ASCII character set in hex order in one field of a two-field buffer area, with the equivalent EBCDIC hex values in a second field in the buffer area. Then, on receipt of an ASCII character its hex value is obtained and matched to the equivalent value in the first field of the buffer area, with the value of the second field containing the equivalent EBCDIC hex value that is then extracted to perform the code conversion.

Members of the IBM PS/2 family and compatible computers use a modified ASCII character set that is represented as an 8-level code. The first 128 characters in the character set, ASCII values 0 through 127, correspond to the ASCII character set listed in Figure 13.18, whereas the next 128 characters can be viewed as an extension of that character set, because they require an 8-bit representation.

Use caution when you transfer IBM PS/2 files; characters with ASCII values greater than 127 will be received in error when they are transmitted using 7 data bits. This is because the ASCII values of these characters are truncated to values in the range 0

Table 13.8
ASCII and EBCDIC Digits Comparison

ASCII			EBCDIC	
Dec	**Oct**	**Hex**	**Hex**	**Digit**
048	060	30	F0	0
049	061	31	F1	1
050	062	32	F2	2
051	063	33	F3	3
052	064	34	F4	4
053	065	35	F5	5
054	066	36	F6	6
055	067	37	F7	7
056	070	38	F8	8
057	071	39	F9	9

to 127 when transmitted with 7 bits from their actual range of 0 to 255. To prevent this problem, initialize your communications software for 8-bit data transfer. However, the receiving device must also be capable of supporting 8-bit ASCII data.

Although conventional ASCII files can be transmitted in a 7-bit format, many word-processing and computer programs contain text graphics represented by ASCII characters whose values exceed 127. In addition, .EXE and .COM files that are produced by assemblers and compilers contain binary data that must also be transmitted in 8-bit ASCII to be accurately received. Whereas most communications programs can transmit 7- or 8-bit ASCII data, many programs may not be able to transmit binary files accurately. This is because communications programs that use the Ctrl+Z character (ASCII SUB) to identify the end of a file transfer misinterpret a group of 8 bits in the .EXE or .COM file being transmitted when the bits have the same 8-bit format as a Ctrl+Z. When Ctrl+Z is detected, the file prematurely closes. To avoid this situation, obtain a communications software program that transfers files by blocks of bits or converts the data into a hexadecimal or octal ASCII equivalent prior to transmission if this type of data transfers will be required.

Error Detection and Correction

As a signal propagates down a transmission medium, several factors can cause errors in reception, including the transmission medium employed and impairments caused by nature and machinery. The transmission medium has a certain level of resistance to current flow that causes signals to attenuate. In addition, inductance and capacitance distort the transmitted signals and there is a degree of leakage that causes a loss in a transmission line due to current flowing across or through insulators, or changes in the magnetic field. Transmission impairments result from numerous sources. First, Gaussian or white noise is always present, because it is the noise level that exists due to the thermal motions of electrons in a circuit. Next, impulse can occur from line hits due to atmospheric static or from poor contacts in a telephone system.

Asynchronous Transmissions

In asynchronous transmissions, the most common form of error control is a single bit, known as a parity bit, used to detect errors. Due to the proliferation of personal computer communications, more sophisticated error detection methods have been developed that resemble the methods employed with synchronous transmission.

Character parity checking, which is also known as *vertical redundancy checking (VRC),* requires an extra bit to be added to each character in order to make the total quantity of 1's in the character either odd or even, depending on whether one is employing odd parity checking or even parity checking. When odd parity checking is employed, the parity bit is set to 1 if the number of 1's in the character's data bits is even; or it is set at 0 if the number of 1's in the character's data bits is odd. When even parity checking is used, the parity bit is set to 0 if the number of 1's in the character's data bits is even; or it is set to 1 if the number of 1's in the character's data bits is odd.

Two additional terms used to reference parity settings are mark and space. When the parity bit is set to a mark condition, the parity bit is always 1, whereas space parity results in the parity bit always set to 0.

For an example of parity checking, consider the ASCII character R, whose bit composition is 1 0 1 0 0 1 0. Because there are three 1 bits in the character R, a 0 bit would be added if odd parity checking is used or a 1 bit would be added as the parity bit if even parity checking is employed. Thus, the ASCII character R would appear as follows:

⌈ data bits ⌉ ↓ parity bit	
1 0 1 0 0 1 0 0	odd parity check
1 0 1 0 0 1 0 1	even parity check

Although parity checking is a simple way to investigate if a single bit error occurred, it can fail when multiple bit errors occur. This can be visualized by returning to the ASCII R character example and examining the effect of two bits erroneously being transformed, as indicated in Table 13.9. Here the ASCII R character has three set bits, and a one bit error could transform the number of set bits to four. Even if parity checking is employed, the received set parity bit would result in the character containing five set bits, which is obviously an error, because even parity checking is employed. Now suppose two bits are transformed in error as indicated in the lower portion of Table 13.9. This means that the character received contains six set bits, which would appear to be correct under even parity checking. Thus, two bit errors in this situation would not be detected by a parity error detection technique.

In addition to the potential of undetected errors, parity checking has several additional limitations. First, when a personal computer is used as an interactive terminal the response to parity errors will vary based on the type of mainframe with which one is communicating. Certain mainframes issue a `Retransmit` message on detection of a parity error. Some mainframes transmit a character that appears as a "fuzzy box" on one's screen in response to detecting a parity error, whereas other mainframes completely ignore parity errors.

When transmitting data asynchronously on a personal computer, the parity on most communications programs can be set to odd, even, off, space, or mark. Off or no parity would be used if the system with which you are communicating does not check the parity bit for transmission errors. No parity would be used when you transmit eight-

Table 13.9 Character Parity Cannot Detect an Even Number of Bit Errors

Action	Example
ASCII character R	1 0 1 0 0 1 0
Adding an even parity bit	1 0 1 0 0 1 0 1
	1
1 bit in error	1 Ø 1 0 0 1 0 1
	1 1
2 bits in error	1 0 Ø 1 Ø 1 0 1

bit EBCDIC or an extended eight-bit ASCII coded data, such as that available on an IBM PS/2 and similar personal computers. Mark parity means that the parity bit is set to 1, whereas space parity means that the parity bit is set to 0.

In the asynchronous communications world, two common sets of parameters are used by most bulletin boards, information utilities, and mainframe computers. The first set consists of seven data bits and one stop bit, using even parity checking; the second set consists of eight data bits and one stop bit using no parity checking. Table 13.10 compares the communications parameter settings of three popular information utilities.

Although visual identification of parity errors in an interactive environment is possible, what happens if you want to transfer a large file over the switched telephone network? For a typical call over the switched telephone network, the probability of a random bit error occurring is approximately 1 in 100,000 bits at a data transmission rate of 1200 bps. If you want to upload or download a 1000 line program containing an average of 40 characters per line, a total of 320,000 data bits are transmitted. During the 4.4 minutes required to transfer this file, you can expect 3.2 bit errors to occur, probably resulting in several program lines being received incorrectly if the errors occur randomly. In such situations, you probably prefer an alternative to visual inspection! Thus, a more efficient error detection and correction method is needed for large data transfers.

Block Checking

Block checking groups data into blocks for transmission. A checksum character is generated and appended to the transmitted block, and the checksum is also calculated at the receiving end, using the same algorithm. If the checksums match, the data block is considered to be received correctly. If the checksums do not match, the block is considered to be in error, and the receiving station can request the transmitting station to retransmit the block. One of the most popular asynchronous block checking methods is included in the *XMODEM protocol,* which is extensively used in personal computer communications. This protocol blocks groups of asynchronous characters together for transmission and computes a checksum that is appended to the end of the block. The checksum is obtained by first summing the ASCII value of each data character in the block and dividing that sum by 255. Then, the quotient is discarded, and the remainder is appended to the block as the checksum. Thus, mathematically the XMODEM checksum can be represented as

Table 13.10 Communication Parameter Settings

	Information Utility		
Parameter	**CompuServe**	**Dow Jones**	**The Source**
Data rate (bps)	300/1200	300/1200	300/1200
Data bits	7/8	8	8
Parity	even/none	none	none
Stop bits	1	1	1
Duplex	full	full	full

$$\text{CHECKSUM} = R\,\frac{\sum_{1}^{128} \text{ASCII Value of Character}}{255}$$

where R is the remainder of the division process. When data is transmitted using the XMODEM protocol, the receiving device performs the same operation on the block being received. This "internally" generated checksum is compared to the transmitted checksum. If the two checksums match, the block is considered to have been received error free. If the two checksums do not match, the block is considered to be in error and the receiving device requests the transmitting device to resend the block. Figure 13.19 illustrates the XMODEM protocol block format. The Start of Header is the ASCII SOH character whose bit composition is 0 0 0 0 0 0 0 1, whereas the 1's complement of the block number is obtained by subtracting the block number from 255. The block number and its complement are contained at the beginning of each block so it's less likely that a line hit at the beginning of the transmission of a block can cause the block to be retransmitted. The XMODEM protocol is examined in more detail in the protocol section of this chapter.

Although the XMODEM protocol significantly reduces the probability of an undetected transmission error occurring in comparison to simple parity checking, it is far from foolproof. As an example of the limitations of XMODEM error detection capability, consider a block consisting of all ones. The ASCII value of the character representing the 1 digit is 49, resulting in the total summed ASCII value of 128 characters in the block becoming 6272. When divided by 255, the quotient is 24 and the remainder of 152 is transmitted as the checksum.

Suppose during the transmission of the XMODEM block two line hits occur, causing one character to be changed from 0110001 to 0110000, whereas a second character is changed from 0110001 to 0110010. In this situation one of the ASCII 1 characters is converted into an ASCII 0 (ASCII value of 48), whereas a second ASCII 1 is converted into an ASCII 2 (ASCII value of 50). Then, during the checksum generation process at the receiving device the total ASCII value of all characters in the block containing two errors is 48 + 50 + (49 × 126), or 6272. When the sum is divided by 255, the remainder of 152 matches the remainder transmitted as the checksum and the block is considered to have been received error free. In spite of being far from foolproof, the XMODEM error detection mechanism is in wide use and can detect approximately 97 percent of randomly occurring transmission errors.

Synchronous Transmission

The majority of error detection schemes employed in synchronous transmission involve geometric codes or cyclic code.

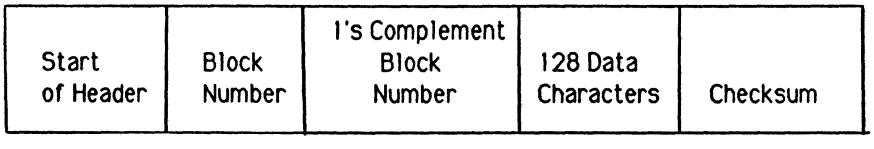


Figure 13.19 XMODEM Protocol Block Format

Geometric codes attack the deficiency of parity by extending it to two dimensions. This involves forming a parity bit on each individual character, as well as on all the characters in the block. Figure 13.20 illustrates the use of block parity checking for a block of 10 data characters. As indicated, this block parity character is also known as the *longitudinal redundancy check (LRC)* character. Geometric codes are similar to the XMODEM error detection technique; they are also far from foolproof. As an example, suppose a 2-bit duration transmission impairment occurred at bit positions 3 and 4 when characters 7 and 9 in Figure 13.20 were transmitted. Here the two 1's in those bit positions might be replaced by two 0's. In this situation, each character parity bit as well as the block parity character fails to detect the errors.

A transmission system using a geometric code for error detection has a slightly better capability to detect errors than the method used in the XMODEM protocol and is hundreds of times better than simple parity checking. Whereas block parity checking substantially reduces the probability of an undetected error in comparison to simple parity checking on a character-by-character basis, other techniques can be used to further decrease the possibility of undetected errors. Among these techniques is the use of cyclic or polynomial code.

When a cyclic or polynomial code error detection scheme is employed, the message block is treated as a data polynomial, D(x), which is divided by a predefined generating polynomial, G(x), resulting in a quotient polynomial, Q(x), and a remainder polynomial, R(x), such that

$$D(x) / G(x) = Q(x) + R(x)$$

The remainder of the division process is known as the *cyclic redundancy check (CRC)* and is normally 16 bits in length or two 8-bit bytes. The CRC checking method is used in synchronous transmission similar to the manner in which the CHECKSUM is employed in the XMODEM protocol previously discussed. That is, the CRC is appended to the block of data to be transmitted. The receiving device uses the same predefined generating polynomial to generate its own CRC based on the received message block

Figure 13.20
VRC/LRC Geometric Code (Odd Parity Checking)

		CHARACTER PARITY BIT
CHARACTER	1	1 0 1 1 0 1 1 0
CHARACTER	2	0 1 0 0 1 0 1 0
	3	0 1 1 0 1 0 0 0
	4	1 0 0 1 0 0 1 0
	5	0 1 1 1 1 0 1 0
	6	1 0 1 0 0 0 0 1
	7	0 1 0 1 1 1 0 1
	8	0 1 1 1 0 0 1 1
	9	1 0 0 0 1 1 0 0
	.	0 1 1 0 1 0 1 1
	.	
BLOCK PARITY CHARACTER (LRC)		1 1 1 0 1 0 0 0

and then compares the "internally" generated CRC with the transmitted CRC. If the two match, the receiver transmits a positive acknowledgment (ACK) communications control character to the transmitting device. The ACK character not only informs the distant device that the data was received correctly but also serves to inform the device that if additional blocks of data remain to be transmitted the next block can be sent. If an error has occurred, the internally generated CRC will not match the transmitted CRC. Then the receiver will transmit a negative acknowledgment (NAK) communications control character that informs the transmitting device to retransmit the block previously sent.

Table 13.11 lists three generating polynomials in common use today. The CRC-16 is based on the American National Standards Institute and is commonly used in the United States. The CCITT CRC is commonly used in transmissions in Europe, whereas the CRC-12 is used with 6-level transmission codes and has been basically superseded by the 16-bit polynomials. The column labeled polynomial in Table 13.11 actually indicates the set bits of the 16-bit or 12-bit polynomial. Thus, the CRC-16 polynomial has a bit composition of 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1.

Protocols

Two types of protocol should be considered in a data communications environment: terminal protocols and data link protocols.

The data link protocol defines the control characteristics of the network and is a set of conventions that govern the transmission of data and control information. A terminal or a personal computer can have a predefined control character or set of control characters unique to the terminal that is not interpreted by the line protocol. This internal protocol can include such control characters as the bell, line feed, and carriage return for conventional teletype terminals; blink and cursor positioning characters for a display terminal; and form control characters for a line printer.

For experimenting with members of the IBM PC Series and compatible computers, try executing the one line BASIC program

```
10 PRINT CHR$(X)"DEMO"
```

substituting different ASCII values for the value of X to see the effect of different PS/2 terminal control characters. As an example, using the value 7 for X, the personal computer will beep prior to displaying the message DEMO, because ASCII 7 is interpreted by the PC as a request to beep the speaker. Using the value 9 for X prints the message DEMO beginning at position 9, because ASCII 9 is a tab character that causes

Table 13.11
Common Generating Polynomials

Standard	Polynomial
CRC-16 (ANSI)	$X^{16} + X^{15} + X^5 + 1$
CRC (CCITT)	$X^{16} + X^{12} + X^5 + 1$
CRC-12	$X^{12} + X^{11} + X^3 + 1$

the cursor to move on the screen 8 character positions to the right. Another example of a terminal control character is ASCII 11, which is the home character. Using the value of 11 for X prints `DEMO` in the upper left-hand corner of the screen, because the cursor is first placed at that location by the home character.

Although poll and select is normally thought of as a type of line discipline or control, it is also a data link protocol. In general, the data link protocol enables the exchange of information according to an order or sequence by establishing a series of rules for the interpretation of control signals that will govern the exchange of information. The control signals govern the execution of a number of tasks that are essential in controlling the exchange of information via a communications facility. Some of these information control tasks are listed in Table 13.12.

Communications Control Characters

Prior to your study of several protocols in more detail, first review the communications control characters in the ASCII character set. These characters were previously listed in Table 13.7, with the two-character designator CC following their meaning. They are reviewed in the order of their appearance in the table.

NUL

As its name implies, the Null character is a nonprintable time delay or filler character. This character is primarily used for communicating with printing devices that require a defined period of time after each carriage return in which to reposition the printhead to the beginning of the next line. Many mainframe computers and bulletin boards operating on personal computers prompt users to `Enter the numbers of nulls`; this is a mechanism to permit conventional terminals, personal computers, and personal computers with a variety of printers to use the system without obtaining garbled output.

SOH

The Start of Heading (SOH) is a communications control character used in several character-oriented protocols to define the beginning of a message heading data block. In synchronous transmission on a multipoint or multidrop line structure, the SOH is followed by an address that is checked by all devices on the common line to ascertain if they are the recipient of the data. In asynchronous transmission, the SOH character can be used to signal the beginning of a filename during multiple file transfers, permitting the transfer to occur without treating each file transfer as a separate communications session. Because asynchronous communications typically involves point-to-point communications, no address is required after the SOH character; however, both

Table 13.12 Information Control Tasks

Connection Tasks	Transmission Tasks
Connection establishment	Transmission sequence
Connection verification	Data sequence
Connection disengagement	Error control procedures

devices must have the same communications software program that permits multiple file transfers in this manner.

STX

The Start of Text (STX) character signifies the end of heading data and the beginning of the actual information contained within the block. This communications control character is used in the bisynchronous protocol, examined later in this chapter.

ETX

The End of Text (ETX) character informs the receiver that all the information within the block has been transmitted. This character is also used to denote the beginning of the block check characters appended to a transmission block as an error detection mechanism. This communications control character is primarily used in the bisynchronous protocol.

EOT

The End of Transmission (EOT) character defines the end of transmission of all data associated with a message transmitted to a device. If transmission occurs on a multidrop circuit the EOT also informs other devices on the line to check later transmissions for the occurrence of messages that could be addressed to them. In the XMODEM protocol, the EOT indicates the end of a file transfer operation.

ENQ

The Enquiry (ENQ) communications control character is used in the bisynchronous protocol to request a response or status from the other station on a point-to-point line or to a specifically addressed station on a multidrop line. In response to the ENQ character, the receiving station responds with the number of the last block of data it successfully received. In a multidrop environment, the mainframe computer polls each device on the line by addressing the ENQ to one particular station at a time. Each station responds to the poll positively or negatively, depending on whether the station has information to send to the mainframe computer.

ACK and DLE

The Acknowledgment (ACK) character is used to verify that a block of data was received correctly. After the receiver computes its own "internal" checksum or cyclic code and compares it to the one appended to the transmitted block, it transmits the ACK character if the two checksums match. In the XMODEM protocol, the ACK character informs the transmitter that the next block of data can be transmitted. In the bisynchronous protocol, the Data Link Escape (DLE) character is normally used in conjunction with the 0 and 1 characters in place of the ACK character. Alternating DLE0 and DLE1 as positive acknowledgment to each correctly received block of data eliminates the potential of a lost or garbled acknowledgement resulting in the loss of data.

NAK

The Negative Acknowledgment (NAK) communications control character is transmitted by a receiving device to request the transmitting device to retransmit the previously

sent data block. This character is transmitted when the receiver's internally generated checksum or cyclic code does not match the one transmitted, indicating that a transmission error has occurred. In the XMODEM protocol, this character informs the transmitting device that the receiver is ready to begin a file transfer operation, as well as to inform the transmitter of any blocks of data received in error.

SYN

The Synchronous Idle (SYN) character is employed in the bisynchronous protocol to maintain line synchronization between the transmitter and receiver during periods when no data is transmitted on the line. When a series of SYN characters is interrupted, it indicates to the receiver that a block of data is being transmitted.

ETB

The End of Transmission Block (ETB) character is used in the bisynchronous protocol in place of an ETX character when data is transmitted in multiple blocks. This character then indicates the end of a particular block of transmitted data.

Bisynchronous Transmission

Among currently used protocols, one of the most frequently used for synchronous transmission is International Business Machines' *BISYNC (binary synchronous communications) protocol.* This protocol is actually a set of very similar protocols that provide rules that transmit binary-coded data synchronously. Although there are numerous versions of the bisynchronous protocol in existence, three versions account for the vast majority of devices operating in a bisynchronous environment. These three versions of the bisynchronous protocol are known as 2780, 3780, and 3270.

The 2780 and 3780 bisynchronous protocols are used for remote job entry communications into a mainframe computer, with the major difference between these versions being that the 3780 version performs space compression, whereas the 2780 version does not. In comparison to the 2780 and 3780 protocols designed for point-to-point communications, the 3270 protocol is designed for operation with devices connected to a mainframe on a multidrop circuit or devices connected to a cluster controller that, in turn, is connected to the mainframe. Thus, 3270 is a poll and select software protocol.

An IBM PS/2 or compatible computer can obtain a bisynchronous communications capability if you install a bisynchronous communications adapter card in the computer's system unit. This card is designed to operate in conjunction with a bisynchronous communications program, which, with the adapter card, enables the PC to operate as an IBM 2780 or 3780 workstation or as an IBM 3270 type of interactive terminal.

The bisynchronous transmission protocol can be used in a variety of transmission codes on a large number of medium- to high-speed equipment. Some of the constraints of this protocol are that it is limited to half-duplex transmission and that it requires the acknowledgment of the receipt of every block of data transmitted. A large number of protocols have been developed due to the success of the BISYNC protocol. Some of these protocols are bit-oriented, whereas BISYNC is a character-oriented protocol. Some permit full-duplex transmission, whereas BISYNC is limited to half-duplex transmission.

Most bisynchronous protocols support several data codes, including ASCII and EBCDIC. Error control is obtained by using a two-dimensional parity check (LRC/VRC)

when transmission is in ASCII. When transmission is in EBCDIC, the CRC-16 polynomial is used to generate a block check character.

Figure 13.21 illustrates the generalized bisynchronous block structure. The start of message control code is normally the STX communications control character. The end of message control code can be the ETX, ETB, or the EOT character; the actual character, however, depends on whether the block is one of many blocks, the end of the transmission block, or the end of the transmission session.

Figure 13.22 illustrates the error control mechanism employed in a bisynchronous protocol to handle the situation in which a line hit occurs during transmission or an acknowledgment to a previously transmitted data block becomes lost or garbled.

In the example on the left portion of Figure 13.22, a line hit occurs during the transmission of the second block of data from the mainframe computer to a terminal or a personal computer. Note that although Figure 13.22 is an abbreviated illustration of the actual bisynchronous block structure and does not show the actual block check characters in each block, in actuality they are contained in each block. Thus, the line hit that occurs during the transmission of the second block results in the "internally" generated BCC being different from the BCC that was transmitted with the second block. This causes the terminal device to transmit an NAK to the mainframe, which results in the retransmission of the second block.

In the example on the right in Figure 13.22, assume that the terminal received block 2 and sent an acknowledgment that was lost or garbled. After a predefined timeout period occurs, the master station transmits an ENQ communications control character to check the status of the terminal. On receipt of the ENQ, the terminal transmits the alternating acknowledgment, currently DLE 1; however, the mainframe was expecting DLE 0. Thus, the mainframe is informed by this that block 2 was never acknowledged and, as a result, retransmits that block.

XMODEM Protocol

The XMODEM protocol originally designed by Ward Christensen has been implemented in many asynchronous personal computer communications programs and a large number of bulletin boards. Figure 13.23 illustrates the use of the XMODEM protocol for a file transfer consisting of two blocks of data. As illustrated, using the XMODEM protocol the receiving device transmits a Negative Acknowledgment (NAK) character to signal the transmitter that it is ready to receive data. In response to the NAK the transmitter

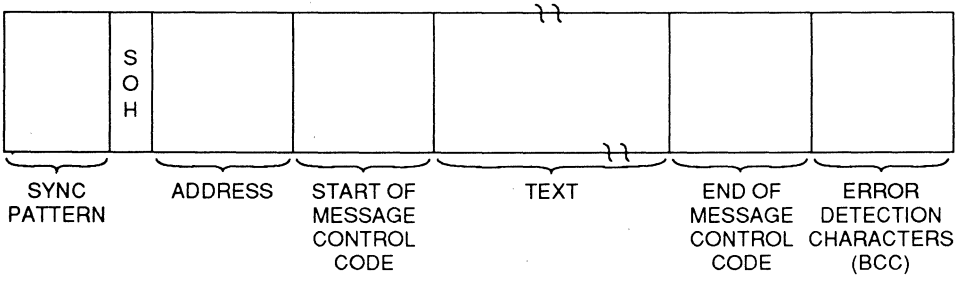


Figure 13.21 Generalized BSC Block Structure

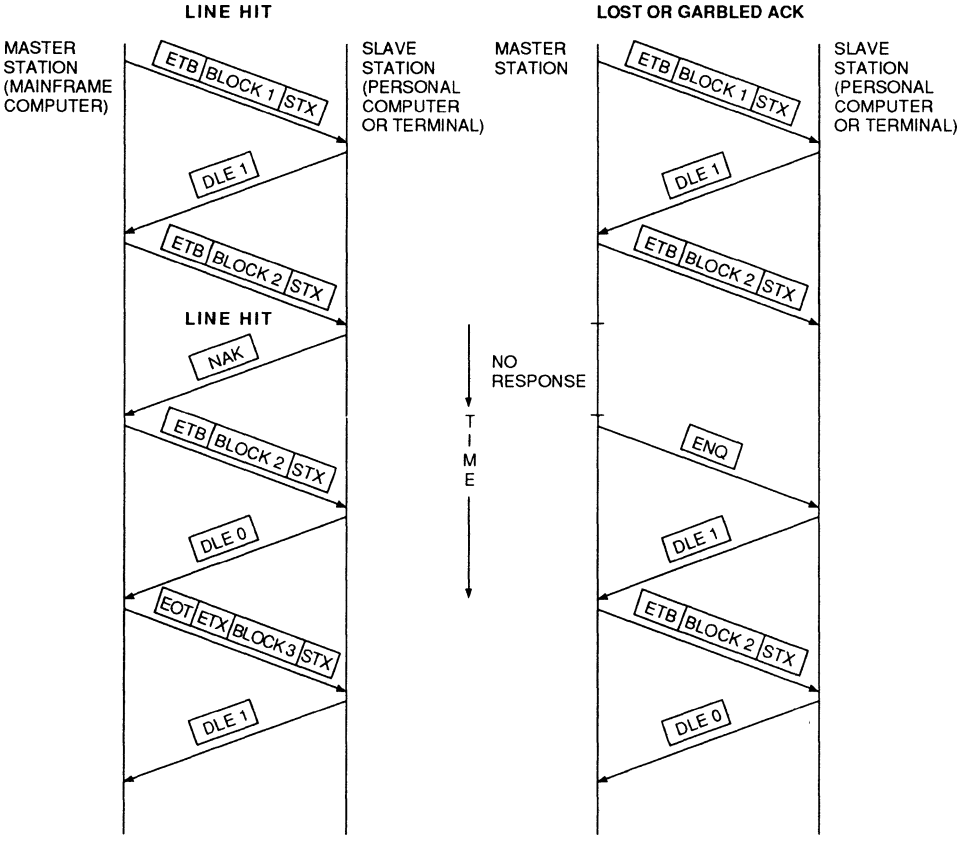


Figure 13.22
BSC Error Control Methods

sends a Start of Header (SOH) communications control character followed by two characters that represent the block number and the one's complement of the block number. Here the one's complement is obtained by subtracting the block number from 255. Next a 128-character data block is transmitted, which, in turn, is followed by the checksum character. As previously discussed, the checksum is computed by first adding the ASCII values of each of the characters in the 128-character block and dividing the sum by 255. Next, the quotient is discarded and the remainder is retained as the checksum.

If the data blocks are damaged during transmission, the receiver can detect the occurrence of an error in one of three ways. If the Start of Header is damaged, it remains undetected by the receiver and the data block is negatively acknowledged. If either the block count or the one's complement field is damaged, it cannot serve as the one's complement of the other. Finally, the receiver computes its own checksum and compares it to the transmitted checksum. If the checksums do not match, this is also an indicator that the transmitted block was received in error. If the two checksums do not match or the SOH was missing or the block count and its complement field are not the one's complement of each other, the block is considered to have been received in

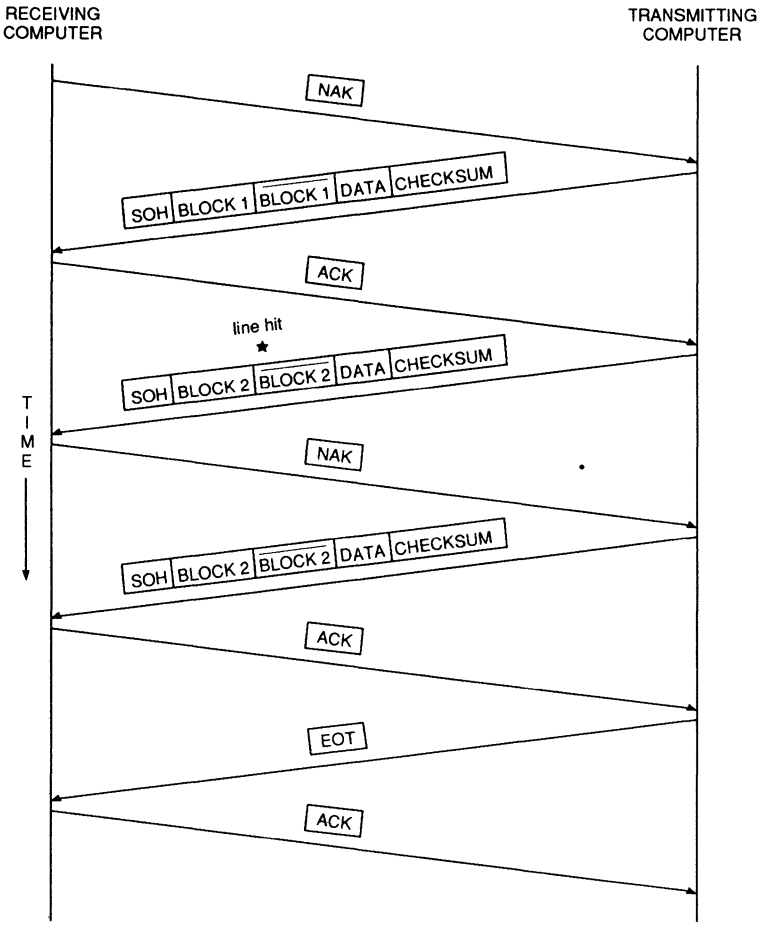


Figure 13.23 XMODEM Protocol File Transfer Operation

error. Then the receiving station will transmit an NAK character that serves as a request to the transmitting station to retransmit the previously transmitted block.

As illustrated in Figure 13.23, a line hit occurring during the transmission of the second block causes the receiver to transmit an NAK and the transmitting device to resend the second block. Suppose more line hits occur that impact the retransmission of the second block. Under the XMODEM protocol the retransmission process is repeated until the block is correctly received or until 9 additional retransmission attempts fail. If due to a thunderstorm or other disturbance line noise is a problem, after 10 attempts to retransmit a block the file transfer process will be aborted. This requires a manual intervention by the sender to restart the file transfer at the beginning and is one of the major deficiencies of the XMODEM protocol.

In spite of the limitations of the XMODEM protocol, it is one of the most popular protocols employed by personal computer users for asynchronous data transfer. This is due to several factors:

- The XMODEM protocol is in the public domain, which means it is readily available at no cost for software developers to incorporate into their communications programs.
- The algorithm employed to generate the checksum is easy to implement using a higher-level language, such as BASIC or Pascal. In comparison, a CRC-16 block check character is normally generated using assembly language.
- The simplistic nature of the protocol is also easy to implement in BASIC or Pascal, which enables many personal computer users to write their own routines to transfer files to and from bulletin boards using this protocol.

Because the XMODEM protocol only requires a 256-character communications receiver buffer, it can be easily incorporated into communications software that operates on personal computer systems with limited memory, such as the early systems that were produced with 64K or less RAM.

Several variations of the original XMODEM protocol have been introduced into the public domain. These modified XMODEM protocols incorporate a true CRC block check character error detection scheme in place of the checksum character, resulting in a much higher level of error detection capability.

Kermit

Kermit was developed at Columbia University in New York City, primarily as a mechanism for downloading files from mainframes to microcomputers. Since its original development, this protocol has evolved into a comprehensive communications system that can be employed for transferring data among most types of intelligent devices. Although the name might imply some type of acronym, in actuality, this protocol was named after Kermit the Frog, the star of the well-known television show, "The Muppets."

Kermit is a half-duplex communications protocol that transfers data in variable sized packets, with a maximum packet size of 96 characters. Packets are transmitted in alternate directions, because each packet must be acknowledged in a manner similar to the XMODEM protocol.

In comparison to the XMODEM protocol, which permits 7- and 8-level ASCII as well as binary data transfers in their original data composition, all Kermit transmissions occur in 7-level ASCII. The reason for this restriction is that Kermit was originally designed to support file transfers to 7-level ASCII mainframes. Binary file transfers are supported by the protocol prefixing each byte whose eighth bit is set by the ampersand (&) character. In addition, all characters transmitted to include 7-level ASCII must be printable, resulting in Kermit transforming each ASCII control character with the pound (#) character. This transformation is accomplished through using the complement of the seventh bit of the control character. Thus, 64 modulo 64 is added or subtracted from each control character encountered in the input data stream. When an eight-bit byte is encountered whose low-order seven bits represent a control character, Kermit appends a double prefix to the character. Thus, the byte 100000001 is transmitted as &#A.

Although character prefixing adds a considerable amount of overhead to the protocol, Kermit includes a *run length compression facility* that may partially reduce the extra overhead associated with control character and binary data transmission. Here, the tilde

(~) character is used as a prefix character to indicate run length compression. The character following the tilde is a repeat count, whereas the third character in the sequence is the character to be repeated. Thus, the sequence ~XA is used to indicate a series of 88 As, because the value of X is 1011000 binary or decimal 88. Through run length compression, the requirement to transmit printable characters results in an approximate 25 percent overhead increase in comparison to the XMODEM protocol for users transmitting binary files. If ASCII data is transmitted, Kermit's efficiency can range from more efficient to less efficient in comparison to the XMODEM protocol, with the number of control characters in the file to be transferred and the susceptibility of the data of run length compression the governing factors in comparing the two protocols. Figure 13.24 illustrates the format of a Kermit packet. The Header field is the ASCII Start of Header (SOH) character. The Length field is a single character whose value ranges between 0 and 94. This one-character field defines the packet length in characters less two, because it indicates the number of characters to include the checksum that follows this field.

The Sequence field is another one-character field whose value varies between 0 and 63. The value of this field wraps around to 0 after each group of 64 packets is transmitted.

The Type field is a single printable character that defines the activity the packet initiates. Packet types include D (data), Y (acknowledgment), N (negative acknowledgment), B (end of transmission or break), F (file header), Z (end of file), and E (error).

The information contents of the packet are included in the Data field. As previously mentioned, control characters and binary data are prefixed prior to their placement in this field.

The Check field can be one, two, or three characters in length, depending on which error detection method is used; this is because the protocol supports three options. A single character is used when a checksum method is used for error detection. When this occurs, the checksum is formed by the addition of the ASCII values of all characters after the Header character through the last data character, and the low-order 7 bits are then used as the checksum. The other two error detection methods supported by Kermit include a two-character checksum and a three-character 16-bit CRC. The two-character checksum is formed similar to the one-character checksum; however, the low-order 12 bits of the arithmetic sum are used and broken into two 7-bit printable characters. The 16-bit CRC is formed using the CCITT standard polynomial, with the high-order 4 bits going into the first character. The middle 6 and low-order 6 bits are placed into the second and third characters, respectively.

By providing the capability to transfer both the filename and contents of files, Kermit provides a more comprehensive capability for file transfers than does XMODEM. In addition, Kermit permits multiple files to be transferred in contrast to XMODEM, which requires the user to initiate each file transfer individually.

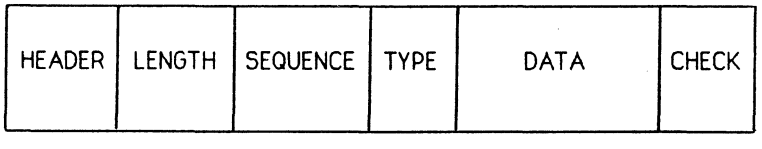


Figure 13.24
The Kermit Packet Format

Bit-Oriented Line Control Protocols

A number of bit-oriented line control procedures have been implemented by computer vendors, based on the International Standards Organization (ISO) procedure known as *High Level Data Link Control (HDLC).* Various names for line control procedures similar to HDLC include IBM's Synchronous Data Link Control (SDLC). The advantages of bit-oriented protocols are threefold:

- Their full-duplex capability supports the simultaneous transmission of data in two directions, resulting in a higher throughput than is obtainable in BISYNC.
- Bit-oriented protocols are naturally transparent to data, enabling the transmission of pure binary data without requiring special sequences of control characters to enable and disable a transparency transmission mode of operation as required with BISYNC.
- Most bit-oriented protocols permit multiple blocks of data to be transmitted. Then, if an error affects a particular block, only that block has to be retransmitted.

SDLC Link Structure

With the SDLC transmission protocol, one station on the line is given the primary status to control the data link and supervise the flow of data on the link. All other stations on the link are secondary stations and respond to commands issued by the primary station.

The vehicle for transporting messages on an SDLC link is called a *frame* and is illustrated in the upper diagram of Figure 13.25.

The SDLC frame contains six fields, wherein two fields serve as frame delimiters and are known as the *SDLC flag.* The SDLC flag has the unique bit combination of 01111110, which defines the beginning and end of the frame. To protect the flag and assure transparency, the transmission device inserts a zero bit any time after a sequence of five one bits occurs to prevent data from being mistaken as a flag. This technique is known as *zero insertion.* The receiver will always delete a zero after receiving five 1's to ensure data integrity.

The address field is an 8-bit pattern that identifies the secondary station involved in the data transfer, whereas the control field can be either 8 or 16 bits long. This field identifies the type of frame transmitted as either an information frame or a command/response frame. The information field can be any length and is treated as pure binary information, whereas the frame check sequence contains a 16-bit value generated using a cyclic redundancy check (CRC) algorithm.

Control Field Formats

The 8-bit control field formats are illustrated in the lower part of Figure 13.25. N(S) and N(R) are the send and receive sequence counts. They are maintained by each station for Information (I-frames) sent and received by that station. Each station increments its N(S) count by one each time it sends a new frame. The N(R) count indicates the expected number of the next frame to be received.

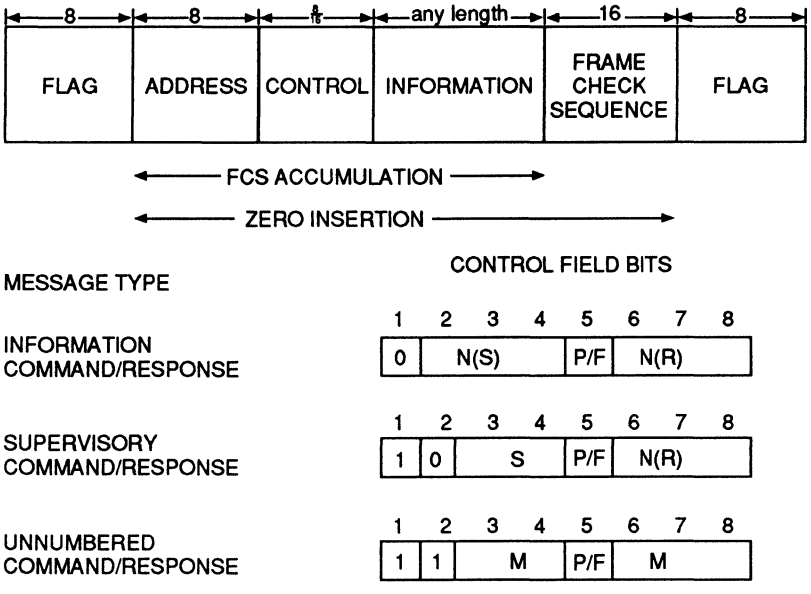


Figure 13.25 SDLC Frame and Control Field Formats

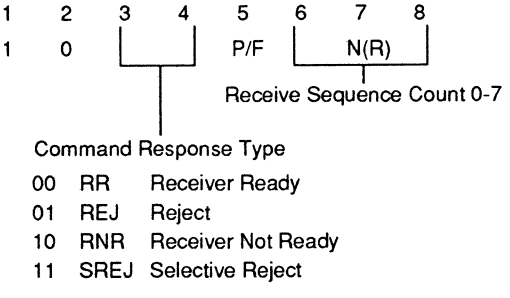


Figure 13.26 Supervisory Control Field

Using an 8-bit control field, the N(S)/N(R) count ranges from 0 to 7. Using a 16-bit control field the count can range from 0 to 127. The P/F bit is a poll/final bit. It is used as a poll by the primary (set to 1) to obtain a response from a secondary station. It is set to 1 as a final bit by a secondary station to indicate the last frame of a sequence of frames.

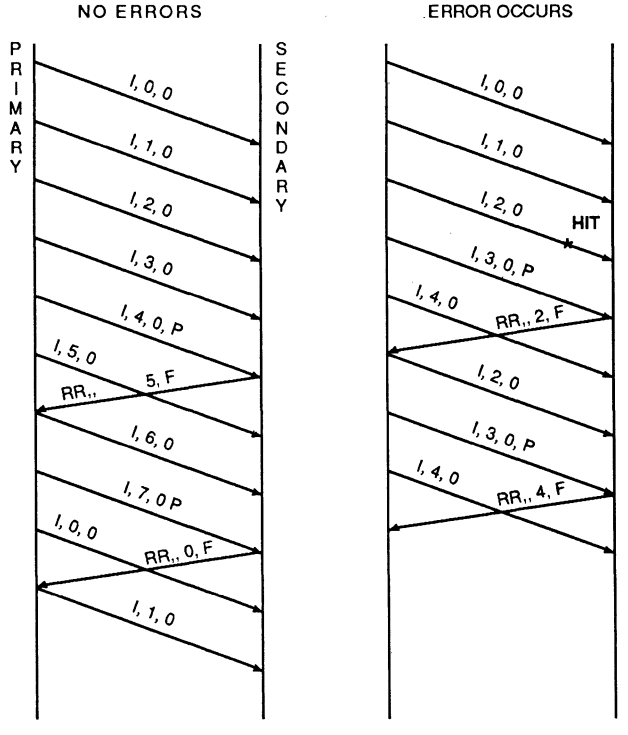
The supervisory command/response frame is used in SDLC to control the flow of data on the line. Figure 13.26 illustrates the composition of the supervisory control

field: supervisory frames (S-frames) contain an N(R) count and are used to acknowledge I-frames, request retransmission of I-frames, request temporary suspension of I-frames, and perform similar functions.

To illustrate the advantages of SDLC over BISYNC transmission, consider the full-duplex data transfer illustrated in Figure 13.27. For each frame transmitted, this figure shows the type of frame, N(S), N(R), and Poll/Final (P/F) bit status.

In the transmission sequence illustrated in the left portion of Figure 13.27 the primary station has transmitted five frames, numbered zero through four, when its poll bit is set in frame four. This poll bit is interpreted by the secondary station as a request for it to transmit its status, and the station responds by transmitting a Receiver Ready (RR) response, indicating that it expects to receive frame five next. This serves as an indicator to the primary station that frames zero through four were received correctly. The secondary station sets its poll/final bit as a final bit to indicate to the primary station that its transmission is completed.

Note that because full-duplex transmission is permissible under SDLC, the primary station continues to transmit information (I) frames while the secondary station is responding to the primary's polls. If an eight-bit control field is used, the maximum frame number that can be outstanding is limited to seven, because 3 bit positions are used for N(S) frame numbering. Thus, after frame number seven is transmitted, the primary station then begins frame numbering again at N(S) equal to zero. Notice that



Format: Type, N(S), N(R), P/F

Figure 13.27 SDLC Full-Duplex Data Transfer

when the primary station sets its poll bit when transmitting frame seven, the secondary station responds, indicating that it expects to receive frame zero. This indicates to the primary station that frames five through seven were received correctly, because the previous secondary response acknowledged frames zero through four.

In the transmission sequence indicated in the right of Figure 13.27, assume a line hit occurs during the transmission of frame two. Note that in comparison to BISYNC, under SDLC the transmitting station does not have to wait for an acknowledgment of the previously transmitted data block; and it can continue to transmit frames until the maximum number of frames outstanding is reached; or, it can issue a poll to the secondary station to query the status of its previously transmitted frames while it continues to transmit frames up until the maximum number of outstanding frames is reached.

Thus, the primary station polled the secondary in frame three and then sent frame four while it waited for the secondary's response. When the secondary's response was received, it indicated that the next frame the secondary expected to receive N(R) was two. This informed the primary station that all frames after frame one would have to be retransmitted. Thus, after transmitting frame four the primary station then retransmitted frames two and three prior to retransmitting frame four.

Note that if selective rejection is implemented, the secondary could have issued a Selective Reject (SREJ) of frame two. Then, on its receipt, the primary station would retransmit frame two and then continue its transmission with frame five. Although selective rejection can considerably increase the throughput of SDLC, even without its use this protocol provides a considerable throughput increase compared to BISYNC.

14 / System/3X and 3270 Networking

This chapter first examines methods to connect IBM PS/2s and compatible personal computers to the IBM System 34, 36, or 38 series of minicomputers. Due to the extensive use of IBM 3270 and third-party equivalent networking devices for communicating with mainframes, the discussion next focuses on this area. After describing the elements of a 3270 Information Display System, the chapter then investigates the methods by which members of the IBM PS/2 family can be integrated into 3270 networks.

Although many older bisynchronous and more modern SDLC operating workstations are not an official part of 3270 networking products, they are included here because of their extensive use with IBM mainframes. After discussing the operation of products that enable PCs to operate as bisynchronous or SDLC workstations, the chapter concludes by discussing two alternatives to hardware products that permit IBM PS/2s to communicate with mainframes as 3270 devices or remote job entry workstations.

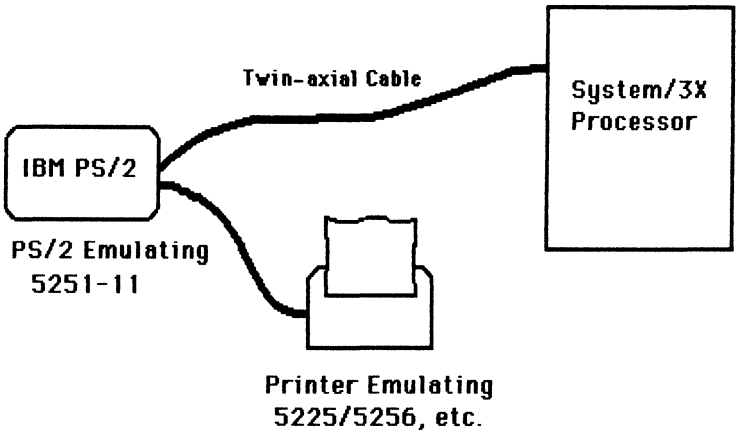
System/3X Connectivity

Due to the large base of installed IBM System 34, System 36, and System 38 minicomputers, IBM—as well as many third-party vendors—developed communications products that enable members of the IBM PS/2 family to communicate with these minicomputers. These products are basically combinations of hardware and software that enable IBM PS/2s to emulate an IBM 5251 local workstation that is normally attached via a twin-axial cable to a System/3X minicomputer.

Local Emulator

Two types of System/3X connectivity can be obtained by selecting appropriate vendor products—local and remote workstation emulation. Figure 14.1 illustrates an IBM PS/2 functioning as a local 5251-11 terminal. To emulate this terminal you install an adapter board in the personal computer. This adapter board attaches directly to an IBM System/3X processor via a twin-axial cable connection. Depending on the degree of vendor hardware and software features included with the IBM 5251 emulation package, the PC user's printer may be supported as an IBM System/3X matrix printer, such as a 4214, 5219, 5224, 5225, or 5256. Other common features offered with IBM 5251 emulation packages include hot key support, file transfer and format translation, record blocking, and security.

Figure 14.1
Local 5251 Emulation



The IBM 5251 emulator hot key is actually a two-keystroke sequence that you use to switch between DOS and multiple System/3X sessions. Concerning file transfer, many 5251 emulation packages translate System/3X formats, such as EBCDIC, Packed, Binary, Alphanumeric, Zoned, and Decimal to PS/2 ASCII, DIF (VisiCalc), WKS (Lotus 1-2-3), Basic, and binary. Similarly, personal computer formats—including those previously mentioned as well as DOS print images—may be converted to one or more System/3X formats.

To enable faster file transfers, many vendors include a record blocking feature. Other vendors incorporate a variety of user IDs, passwords, and file and record security features to both protect and enable users to maintain control of their System/3X database resources.

Currently, IBM markets two types of local emulation products designed to provide connectivity between a PS/2 and a System/3X. IBM's Enhanced 5250 Display Station Emulation Convenience Kit Version 2.12 supports PC bus computers to include the IBM PS/2 Model 30. This kit includes the Enhanced 5250 Display Station Emulation Program Version 2.12, an Enhanced 5250 Display Emulation Adapter that is designed for use in a PC bus personal computer, and an integrated cable assembly to connect the rear of the adapter card to a cable from a System/3X. Using this kit your personal computer can emulate an IBM 5291, 5292-1, or 5292-2 workstation, obtaining the ability to execute System/3X sessions and PS/2 applications concurrently and switch from one to the other by pressing a hot key. Up to two host sessions can be active concurrently with one PS/2 session.

The second type of local emulator marketed by IBM for System/3X connectivity includes the vendor's System/36/38 Workstation Emulation Adapter/A, which is designed to be installed in a Micro Channel PS/2. When used with the Enhanced 5250 Display Station Emulation Program, this combination of hardware and software provides the same functionality as the convenience kit.

Remote Emulator

A remote emulator package for System/3X access enables an IBM PS/2 or compatible computer to function as a 5251 workstation via a dial-up or leased line connection to

the System/3X processor. Several types of 5251 emulation packages are currently marketed by vendors, with major differences in the inclusion or exclusion of an autodial modem on the emulator adapter card and the type of emulation supported.

Some vendors now include asynchronous or synchronous 2400 or synchronous 4800 bps modems on their 5251 remote adapter cards. Because it replaces an external unit and does not require the casing of a standalone device, such a modem saves desk space as well as costs less than an adapter card and standalone modem. Although most remote emulation packages turn the PC into a remote 5251 workstation, as illustrated in the top portion of Figure 14.2, other emulation packages help the PS/2 emulate an IBM 5294 remote controller. Then, other personal computers at the remote location can be connected to the PS/2 emulating the controller, as illustrated in the lower portion of Figure 14.2.

When a 5251 remote emulation package is obtained, the IBM SNA/SDLC protocol handles data transmission from the remote personal computer to the System/3X minicomputer. Although many vendors offer the same adapter boards due to *original equipment manufacturer (OEM)* agreements between a few board manufacturers and a large number of retailers, software features can vary to a significant degree. This is because vendors supply different types of software modules to retailers that bundle the software with the hardware as a system. Based on this, you should probably concentrate a higher percentage of the time spent evaluating 5251 emulation packages on the software part of the package.

The IBM 3270 Information Display System

The IBM 3270 Information Display System describes a collection of products ranging from *display stations* (terminals) with keyboards and printers that communicate with

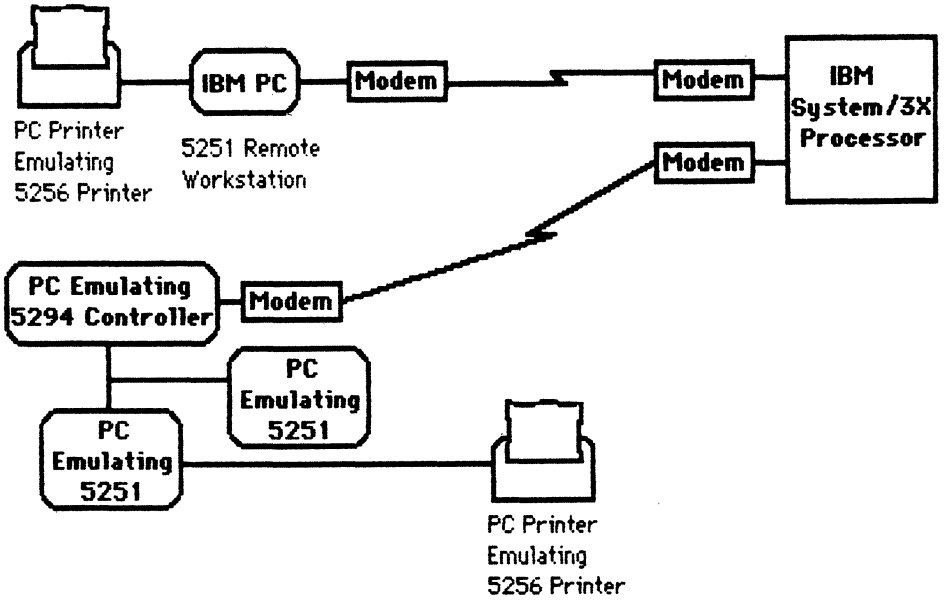


Figure 14.2
Remote 5251
Emulation

mainframe computers to several types of cluster controllers, including a now-obsolete member of the IBM PC series known as the 3270 PC.

First introduced in 1971, the IBM 3270 Information Display System was designed to extend the processing power of the mainframe computer, commonly called a *host* by IBM, to locations remote from the computer room. Controllers, which are called *control units* by IBM, were made available to economize on the number of lines required to link display stations to mainframe computers. Typically, a number of display stations are connected to a control unit on individual cables and the control unit, in turn, is connected to the mainframe via a single cable. Both local and remote control units are offered, with the key differences between the two pertaining to the method of attachment to the mainframe computer and the use of intermediate devices between the control unit and the mainframe.

Local control units are usually attached to a channel on the mainframe, whereas remote control units are connected to the mainframe's front-end processor, also known as a *communications controller* in the IBM environment. Because a local control unit is within a limited distance of the mainframe, no intermediate communications devices, such as modems, are required to connect a local control unit to the mainframe. In comparison, a remote control unit can be located in another building or in a different city. The remote control unit normally requires the use of intermediate communications devices, such as a pair of modems, for communications to occur between the control unit and the communications controller. The relationship of local and remote control units to display stations, mainframes, and a communications controller is illustrated in Figure 14.3.

Control Unit Operation

The control unit polls each connected display station to ascertain if the station has data stored in its transmit buffer. If the station has data in its buffer, the station transmits the data to the control unit when the station is polled. The control unit then formats the data with the display station's address, adds the control unit's address and other pertinent information, and transmits it in a synchronous data format to the communications controller or to the I/O channel on the mainframe, depending on the method used to connect the control unit to the mainframe.

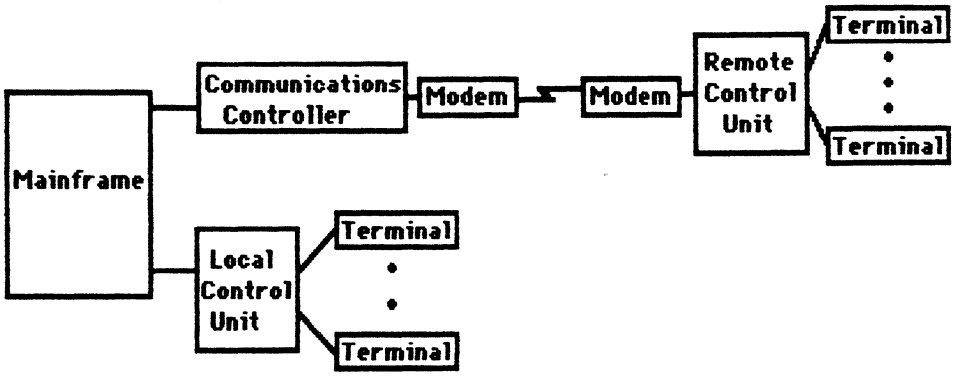


Figure 14.3
Relationship of 3270 Information Display Products

3270 Protocols

Two protocols are supported by IBM to connect 3270 devices to a mainframe. The original protocol used with 3270 devices that is still widely used today is the *byte-oriented bisynchronous protocol,* often referred to as *3270 bisync* or *BSC.* In the late 1970s, IBM introduced its *Systems Network Architecture (SNA),* which is basically an architecture that permits distributed systems to be interconnected based on a series of conventions. Conventions include a bit-oriented protocol for data transmission known as *Synchronous Data Link Control,* or *SDLC.* Thus, communications between an IBM mainframe and the control units attached to the communications controller are either BSC or SDLC, depending on the type of control units obtained and the configuration of the communications controller that is controlled by software.

Types of Control Units

Control units currently marketed support up to 8, 16, or 32 attached devices, depending on the mode. The now obsolete IBM 3276 control unit (still in common use) supports up to 8 devices, whereas the more modern IBM 3174 and 3274 control units can each support 16 or 32 attached devices. Older control units—such as the 3271, 3272, and 3275, which only operate bisynchronously—have largely been replaced by the 3174 and 3274 control units. Certain models of the 3274 and all 3174s are "soft" devices that can be programmed with a diskette to operate with the originally developed bisynchronous protocol or with the newer SDLC protocol.

Devices such as display stations and printers are normally attached to each control unit via coaxial cable. Thus, under this design philosophy every display station must first be connected to a control unit before it can access a mainframe application written for a 3270 type terminal. This method of connection excluded the easy access by dial-up terminals to 3270 type applications and resulted in numerous third-party vendors marketing devices used by lower-cost asynchronous ASCII terminals to attach to 3270 networks. In late 1986, IBM introduced its 3174 Subsystem Control Unit. This control unit can be used to connect terminals via standard coaxial cable, shielded twisted-pair wire and telephone type twisted-pair wire. Other key features of this controller include an optional protocol converter that can support up to 24 asynchronous ports and the ability of the controller to be attached to IBM's Token Ring Local Area Network.

Before you explore the methods by which personal computers can be connected to IBM control units, review the classes of contemporary 3270 type terminals, because techniques developed to integrate personal computers into a 3270 network revolve around making the PC function as a particular 3270 type terminal. In addition, you will examine the IBM 3270 PC, which was specifically developed to support multiple host computer sessions for an IBM mainframe computer. By using the preceding as a foundation of knowledge concerning 3270 terminal and 3270 PC operations, you can then master the techniques that enable conventional personal computers to communicate with mainframes as 3270 type devices.

Terminal Displays

Currently, IBM 3270 terminals fall into three display classes: monochrome, color, and gas plasma. Members of the *monochrome display* class include the 3278, 3178, 3180,

3191, and 3193 terminals. The 3278 is a large, bulky terminal that easily covers a significant portion of a desk. It has basically been replaced by the 3178, 3180, 3191, and 3193 display stations that are lighter, more compact, and less expensive. The 3279 color display station was similarly replaced by the 3179 and 3194 that are lower cost and more compact color display terminals. The last class of display stations, gas plasma display, consists of the 3270 flat panel display.

The physical dimensions of a 3270 screen may vary by class and model within the class. As an example, the 3178 and 3278 Model 2 display stations have a screen size of 24 rows by 80 columns, whereas the 3278 Model 3 has a screen size of 32 rows by 80 columns, and the 3278 Model 4 has a screen size of 43 rows by 80 columns.

Table 14.1 lists the family of terminals marketed for use with the IBM 3270 Information Display System. Note that the 3179 and 3180 display stations can also be used with IBM System/36 and System/38 minicomputers. In addition, the 3193 and 3194 terminals can support the display of up to four host sessions when connected to the 3174 controller.

Each 3270 screen consists of fields that are defined by the application program connected to the display station. Attributes sent by the application program further define the characteristics of each field as indicated in Table 14.2. As a minimum, any

Table 14.1
IBM Display Stations

Model Number	Display Type	Screen (inches)
3178	monochrome	12
3179	color	14
3180	monochrome	15
3191	monochrome	12
3193	monochrome	15
3194	color	14

Table 14.2
3270 Terminal Field Characteristics

Field Characteristic	Result
highlighted	Field displayed at a brighter intensity than normal intensity field.
nondisplay	Field does not display any data typed into it.
protected	Field does not accept any input.
unprotected	Field accepts any data typed into it.
numeric-only	Field only accepts numbers as input.
autoskip	Field sends the cursor to the next unprotected field after it is filled with data.
underscoring	Underlines characters.
blinking	Causes characters in field to blink.

technique used to enable a personal computer to function as a 3270 display station requires the computer to obtain the field attributes listed in Table 14.2.

3270 Keyboard Functions

In comparison to the keyboard of a member of the IBM PS/2 family, a 3270 display station contains approximately 40 additional keys. When pressed, the keys perform functions unique to the 3270 terminal environment. Table 14.3 lists the more common 3270 keys that differ from the function of keys on an IBM PS/2 keyboard.

Most, if not all, of the 3270 keyboard functions may be required to successfully use a 3270 application program. Thus, the codes generated on a personal computer keyboard must be converted to appropriate 3270 keyboard codes to enable a PS/2 to be used as a 3270 terminal. Because there are fewer keys on a personal computer keyboard, a common approach to most emulation techniques is to use a two- or three-key sequence on the PS/2 keyboard to represent many of the keys unique to a 3270 keyboard.

Emulation Considerations

In addition to converting keys on an IBM PS/2 keyboard to 3270 keyboard functions, 3270 emulation requires the computer's screen to function as a 3270 display screen. The 3270 display terminal operates by displaying an entire screen of data in one operation, then waits for the operator to signal that he or she is ready to proceed with

Table 14.3 Common 3270 Keys Differing in Functionality from IBM PS/2 Keyboard Keys

Key(s)	Function
CLEAR	Erases screen except for characters in message area, repositioning cursor to row 1, column 1.
PA1	Transmits a code to the application program that is often interpreted as a request to redisplay the screen or to clear the screen and display additional information.
PA2	Transmits a code to the application program that is often interpreted as a request to redisplay the screen or to clear the screen and display additional information.
PFnn	Twenty-four program function keys on a 3270 terminal are defined by the application program in use.
TAB	Moves the cursor to the next unprotected field.
BACKTAB	Moves the cursor to the previous unprotected field.
RESET	Disables the insert mode.
ERASEEOF	Deletes everything from the cursor to the end of the input field.
NEWLINE	Advances the cursor to the first unprotected field on the next line.
FASTRIGHT	Moves the cursor to the right 2 characters at a time.
FASTLEFT	Moves the cursor to the left 2 characters at a time.
ERASE INPUT	Clears all the input fields on the screen.
HOME	Moves the cursor to the first unprotected field on the screen.

the next screen of information. This "full-screen" operation mode is exactly the opposite of TTY emulation, where a terminal operates "on a line by line" basis. A key advantage of full-screen editing is the ability of the operator to move the cursor to any position on the screen to edit or change data. Thus, to use an IBM PS/2 as a 3270 display station, the transmission codes used to position the 3270 screen and effect field attributes must be converted to equivalent codes recognizable by the PS/2.

3270 PC

The IBM 3270 PC functions both as a 3270 type terminal and as a personal computer. Although its system unit is identical to that of the PC XT, its functionality is obtained from a series of adapter cards that are installed into the system expansion slots in the system unit. Additional significant differences between the 3270 PC and other members of the PC series include its keyboard, display screen, operating system, method of mainframe connection, and price.

The 3270 PC's keyboard has 39 keys more than a conventional IBM PC keyboard, which are used to provide access to 3278 terminal functions. Incorporating these keys on the 3270 PC keyboard eliminates the multiple key sequences used by many emulation programs to obtain equivalent 3278 terminal key representations. Some vendors, such as Digital Communications Associates, market a 3270 PC compatible keyboard that can be used to eliminate the frequent annoyance of trying to remember which key sequence on a conventional PC keyboard is used by an emulation program to represent a particular 3278 terminal key.

Although the 3270 PC supports most conventional versions of DOS, its primary operating system is a proprietary Control Program that is designed to facilitate the connection and operation of the 3270 PC with a mainframe computer. Under the 3270 PC Control Program, your system can be connected to four separate mainframe sessions at one time. Multiple sessions are managed by the 3270 PC employing a windowing display format, so you can partition the screen into as many as seven windows. Up to four windows can be used for mainframe computer sessions, whereas one window can be used for DOS and the remaining two windows can be used as notepads. Another unique feature of the 3270 PC is its file transfer software, which operates in conjunction with special software that must be installed on the mainframe computer.

To transmit a file to the mainframe, you use the SEND command, specifying the physical storage location of the file on the 3270 PC and its name, as well as the mainframe filename it should be stored under. Because IBM mainframes normally store data files in EBCDIC, the 3270 PC files are translated into EBCDIC for storage on the mainframe. Similarly, a RECEIVE command permits files on the mainframe to be converted into ASCII and downloaded onto the 3270 PC.

Designed to operate as a 3270 type terminal, the 3270 PC must be interfaced via a coaxial connection to a control unit. Then, the control unit can be directly attached to the mainframe (channel attached) or connected to the communications controller that in turn is directly connected to the mainframe computer. This method of connection precludes using the 3270 PC in a dial-up mode to directly access a mainframe.

Although you could originally purchase a 3270 PC to obtain an interface into the world of 3270 networking, two major factors contributed to the tremendous growth in IBM and third-party products that provide conventional personal computers with similar

networking capability. These factors are price and performance. From a cost perspective, it was often less expensive to obtain an IBM PC or compatible computer and, through the use of IBM or third-party products, obtain a hardware configuration that permitted the personal computer to access a mainframe similar to a 3270 type terminal. Regarding performance, the variety of hardware and software products marketed by IBM and third-party vendors enables users to consider many alternative methods for attaching PCs to 3270 type networks, with varying levels of performance and cost associated with each method. Due to this, the 3270 PC never experienced significant market success and is now an obsolete product that is no longer manufactured.

3270 Connection Methods

You can use numerous methods to attach non-3270 type terminal devices (including IBM PS/2s) to a 3270 network. Such methods basically fall into four defined categories: protocol converters, terminal emulators, micro/host software emulators, and specialized facilities offered by some packet networks. Each category has many advantages and disadvantages, but the actual method employed is based on the existing network architecture of the organization, including control unit and personal computer locations and the requirements for linking PCs to the mainframe computers. So bear in mind that the examples discussed in this chapter may not be applicable to all situations due to differences in the distribution of control units and personal computers among networks. Instead, the material presented in the remainder of this chapter should be used as a guide to the various approaches for connecting personal computers into a 3270 type network.

Protocol Converters

As the name implies, a protocol converter converts dissimilar devices that communicate in two different protocols, enabling such devices to achieve communications compatibility. In an IBM 3270 networking environment, a protocol converter is normally designed to be interfaced between the communications control unit or front-end processor and a terminal device, or it can be installed directly into the system unit of a personal computer. The protocol converter is then either directly cabled to the front-end processor if the personal computer is located close to the mainframe or via a pair of modems if the PC is at a remote location. Another networking option commonly employed is to have personal computers transmit data via the public switched telephone network (PSTN) or a value-added carrier to obtain access to a protocol converter that in turn is connected to the communications controller.

Most protocol converters are designed to appear to the front end as a 3270 type control unit on one side of the device and supporting asynchronous ASCII data input at the other end of the device. Figure 14.4 illustrates the use of protocol converters at both a local and remote location with respect to the corporate mainframe computer. Note that personal computers can access the protocol converter via a direct cable connection, over the PSTN into a modem that in turn is connected to the protocol converter, or using a packet network as a data transportation highway to the location where the protocol converter is installed.

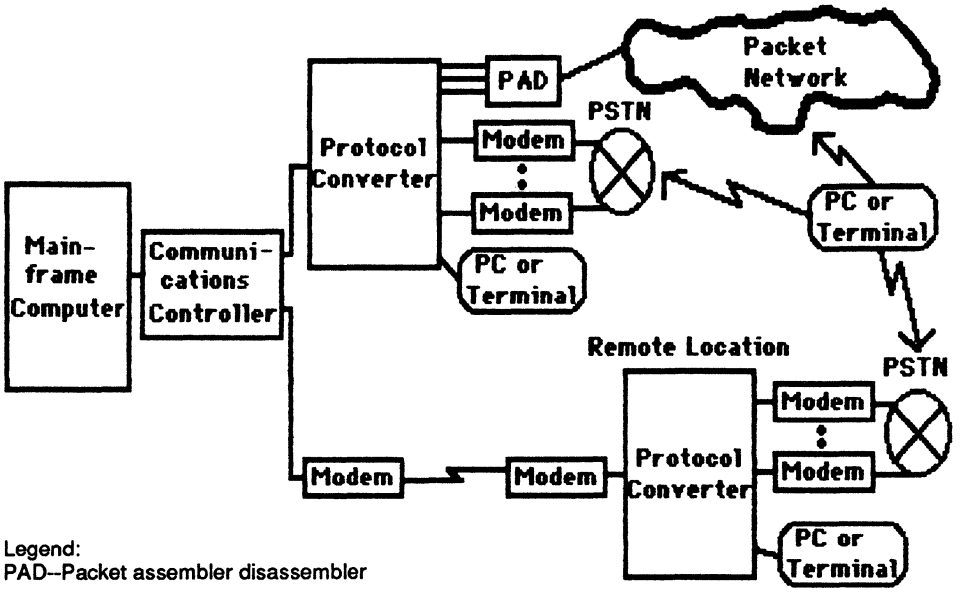


Figure 14.4
Employing Protocol Converters

The protocol converter translates the 3270 screen formats into the character sequences recognized by the personal computer. In addition to performing this data formatting conversion, the protocol converter performs several additional functions, including data link control conversion and transmission protocol conversion. Here the data link control conversion is simply mapping the communications line-handling sequence of the personal computer to that of a 3270 type terminal. The transmission protocol processing is the conversion of the line by line data flow through the PS/2's communications buffer into data blocks with appropriate addressing and error detection that match the 3270 communications protocol supported by the communications controller.

Normally, the intelligence of the protocol converter is a ROM cartridge. PS/2s communicating with the protocol converter use a program that works in conjunction with the ROM cartridge to obtain appropriate screen translations. Among the most popular types of personal computer software used for communicating with protocol converters are DEC VT100 and IBM 3101 terminal emulators due to the popularity of these terminals as well as the standardization of the DEC VT100 screen codes by the American National Standards Institute.

Because a 3270 type control unit is designed to have terminals permanently connected to the device, a logical question that might arise is how protocol converters support access through value-added carriers or the PSTN. Whenever someone signs off a mainframe session, the protocol converter continues to respond to communications controller polls, so that device functions as though terminals are inactive but attached to each protocol converter port. Then, after someone accesses the protocol converter via a value-added carrier or the PSTN, the protocol converter transmits an appropriate code to the communications controller to indicate that an active terminal requiring servicing is there.

Control Unit Emulation

When the protocol converter emulates the operation of a control unit, the process is known as *controller* or *control unit emulation.* The previous discussion of protocol converters focused on standalone devices. A second category of protocol converters is installed in the system unit of a PC and in effect converts the personal computer into a control unit.

One example of a protocol converter installed within the system unit of an IBM PC or a PC bus PS/2 that operates as a controller emulator is the Persyst DCP/88 hardware adapter. It is used in conjunction with the firm's PC/3270 series software. This combination of hardware and software converts an IBM PC or PC bus PS/2 into a miniature 3274 control unit, which is then capable of supporting up to four additional devices, including three other personal computers operating as 3278 type terminals and one high-speed printer operating at 600 lines per minute. This is illustrated in Figure 14.5. The Multiple Protocol Communications Controller illustrated in Figure 14.5 is another Persyst adapter board installed in a PC or PC bus PS/2 system unit. This adapter enables a PC to emulate a 3278/9 terminal supporting 3270 bisynchronous communications.

In addition to enabling individual PCs to be connected to the company's DCP/88, the MPC can be installed in a PC to provide it with the capability to be connected from a remote location to a mainframe via the mainframe's communications controller, as illustrated in Figure 14.6. Using the Persyst MPC board illustrated in Figure 14.5, a device functions as a terminal emulator, because it connects a PC to a control unit that,

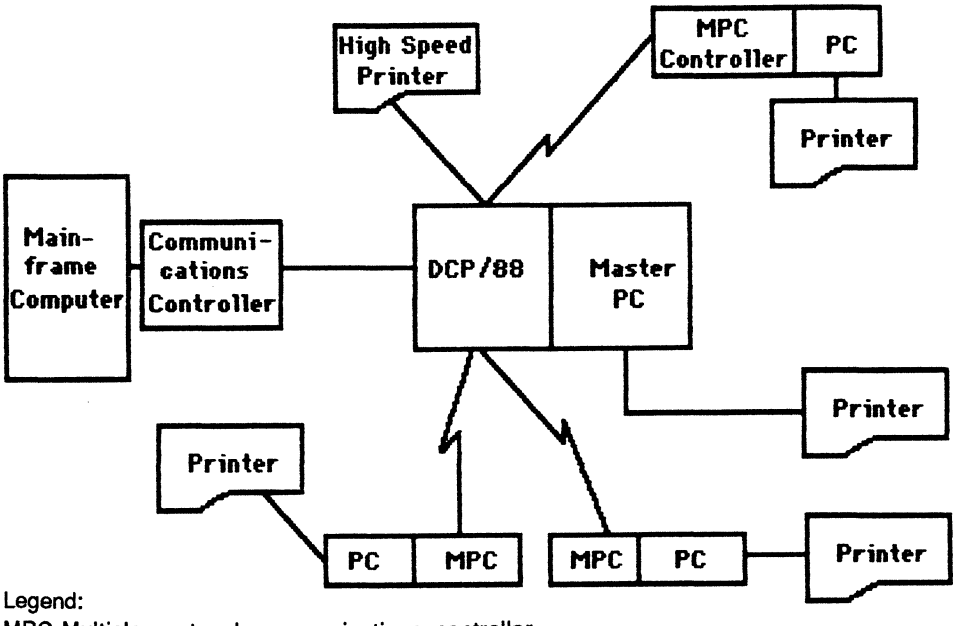


Figure 14.5
Persyst DCP/88 Converts a PC into a Control Unit

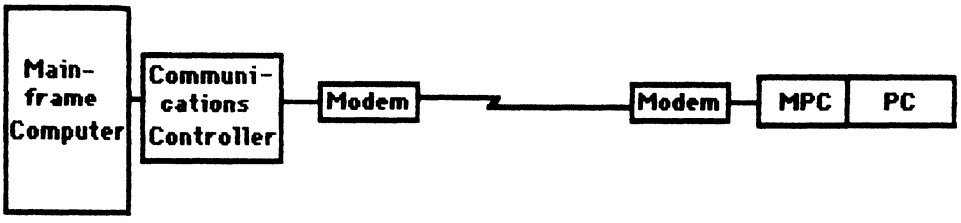


Figure 14.6
Remote Operations of Persyst MPC Board

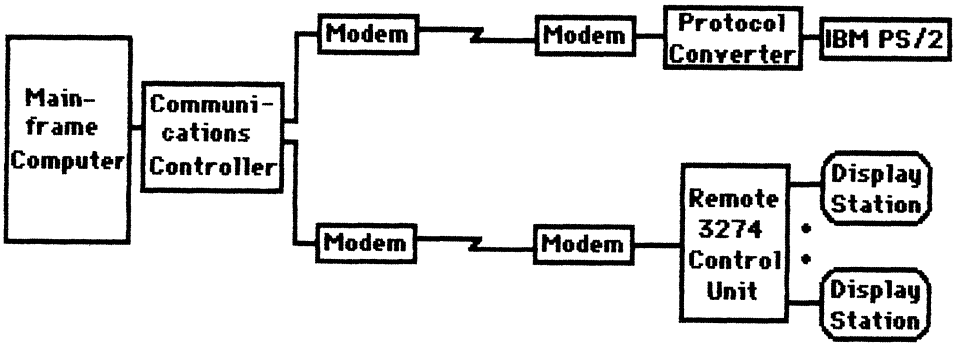


Figure 14.7
Controller Emulation May Not Be Cost Effective

in this particular instance, is the vendor's DCP/88 product. When the MPC board is used to connect a personal computer directly to the communications controller, it functions as a combined control unit and terminal emulator. This is because all terminals in a 3270 network must be connected via control units to the mainframe or to the communications controller. Thus, a single personal computer at a remote location must appear to the communications controller as a terminal connected to a control unit.

Although a multiline protocol converter operating as a control unit emulator is an economical method to link a number of colocated PS/2s onto a common line to the mainframe, the economics of this approach may deteriorate under different network topologies. One example of a network topology that may be economically unsuitable for control unit emulation is illustrated in Figure 14.7. Here a remote office has an existing 3274 control unit connected to several IBM or plug compatible display stations and one IBM PC. To use a protocol converter that functions as a control unit emulator, a separate communications line must be installed between the mainframe and the remote office, as well as an additional pair of modems. In addition, another port on the communications controller is required to support the additional line, further increasing the cost of linking the PS/2 to the corporate mainframe in this type of situation.

Terminal Emulation

A terminal emulator economically attaches PS/2s to control units. Terminal emulation can occur in several ways. First, you can use a protocol converter that functions as a terminal emulator. This type of device is connected between the personal computer and the control unit, making the PS/2 appear as a 3278 type terminal to the control unit.

The protocol converter illustrated in Figure 14.8 is a standalone device that is connected to the personal computer on one end and cabled to the control unit on the other end.

A second and more popular type of terminal emulator is a printed circuit board that is installed into one of the expansion slots of an IBM PS/2. The board provides the personal computer with a 3278/9 screen display so the computer user can access mainframe full-screen applications. When appropriate diskette-based software is added, your PS/2 can transfer data to and from the mainframe.

IBM and several third-party vendors have introduced terminal emulator products so a member of the IBM PS/2 family can connect via a coaxial cable to a local or remote control unit. These products consist of an adapter board inserted into the system unit of the PC and appropriate software that permits the PS/2 to function as a 3278/9 type terminal.

There are several IBM products you can consider to connect your PS/2 to a mainframe through a control unit. The hardware required to obtain a terminal emulation capability that provides the connectivity between your PS/2 and a control unit depends on your computer's bus structure. For IBM PC bus based personal computers, such as the PS/2 Model 30, IBM markets a 3278/3279 Emulation Adapter. Designed for insertion in a PC bus personal computer, its use with appropriate software enables you to transport host files to your personal computer, modify the files on your PS/2, and upload the files back to the mainframe. For Micro Channel bus PS/2s, IBM markets its 3270 Connection, which is an adapter board designed for installation in PS/2 computers that have the Micro Channel bus.

Both adapter cards require appropriate software to obtain PC to mainframe connectivity. You can use the IBM PC 3270 Emulation Program Entry Level Version, the IBM PC 3270 Emulation Program Version 3.0, or the IBM 3270 Workstation Program with both adapters.

The IBM PC 3270 Emulation Program Entry Level package supports access from your PS/2 to mainframe computer programs as if your PS/2 were a 3278 or 3279 type terminal. Included in the software is the capability to use SEND and RECEIVE commands from within DOS to transfer files between your PS/2 and the mainframe once IBM's host-supported File Transfer Program is installed on your mainframe computer. Other functionality obtained from this software includes the ability to redefine keys on the PS/2 keyboard and the ability to select your color scheme for host computer sessions.

The IBM PC 3270 Emulation Program Version 3 is the more sophisticated of the three programs you can use to obtain host connectivity. This product provides 3270 emulation connectivity similar to the Entry Level package, with the added capability

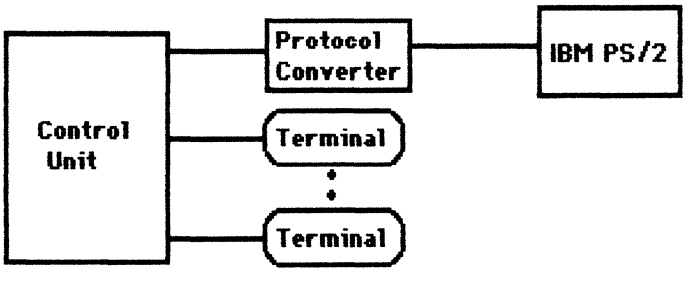


Figure 14.8
Protocol Converter Functioning as Terminal Emulator

to save 3270 screens and route print jobs directly to a printer or disk. Due to a memory conflict between the Entry Level program and IBM's PC Local Area Network program, to communicate on a local area network using IBM's LAN software and access a mainframe directly you must use the 3270 Emulation Program Version 3 software. The latter was written specifically to permit 3270 access to coexist with a local area network session and permits your PS/2 to function as one of four types of 3270 workstations—standalone, gateway, gateway with network station, or network station. Here a gateway permits the PS/2 to act as a controller for other personal computers on a local area network that require access to a host computer, converting the physical communications between host and network station. When configured as a gateway with network station, your PS/2 can function as both a gateway to a host and as a network station on a local area network.

The third IBM software product—the IBM 3270 Workstation Program—helps you use your PS/2 to communicate on a local area network and access a host computer. Use this program to communicate with a mainframe via a control unit as well as to operate the IBM PC Local Area Network Program concurrently. Table 14.4 summarizes the features of each program.

A few of the more popular third-party terminal emulator products include Digital Communications Association's IRMA decision support interface, CXI's 3270-PC Connection, Forte Data Systems' 3270/PC package, and the previously referenced Persyst MPC adapter.

DCA's IRMA

IRMA and IRMA/2 are printed circuit boards you install in system expansion slots of PC bus and Micro Channel-based PS/2 computers, respectively. Each board provides the PS/2 user with access to the mainframe by emulating a 3278 or 3279 type terminal. Purchasers are provided with file transfer software that uploads and downloads files between the PS/2 and the mainframe.

Software supplied by DCA with the IRMA terminal emulator adapter board includes file transfer programs to access IBM VM/CMS and MVS/TSO host computer environments. The file transfer program for use with a VM/CMS host environment operates

Table 14.4
IBM 3270 Connectivity Software

Program	Function
Entry Level	Provides direct connection to host computer. Does not support concurrent use of IBM PC Local Area Network Program.
Emulation Program Version 3.0	Provides direct connection to host computer and indirect connection to host from other local area network users. Supports concurrent use of IBM PC Local Area Network Program.
Workstation Program	Provides direct connection to host computer. Supports concurrent use of IBM PC Local Area Network Program.

with the XEDIT editor, whereas the second file transfer program uses the EDIT function of TSO. Due to their reliance on the use of editor facilities, file transfers using this software are relatively slow; after one line of data is transmitted to the mainframe the program waits for the host's editor response prior to transmitting the next line of data. Due to this, several third-party vendors have introduced specialized programs that operate on the mainframe and the PC and result in a significant increase in file transfer throughput. Recognizing the slowness of their conventional file transfer, DCA now provides IRMA users without cost two high-speed file transfer programs that operate on mainframes running VM/CMS, MVS/SP/TSO, or MVS/XA/TSO operating system environments. These programs used in conjunction with software operating on the personal computer can result in file transfer operations approaching a 25,000 to 35,000 character per minute data transfer rate.

Conventional File Transfer

Prior to performing a file transfer, the PS/2 operator uses a DCA software program called E78 (if emulating an IBM 3278 terminal) in conjunction with the IRMA adapter board to enable full-screen access to the mainframe via a control unit. Assuming that the user's host environment is MVS/TSO, after log-on the PC operator accesses TSO and receives a READY prompt. Next, by pressing at the same time the two shift keys on the PC's keyboard the E78 program is exited and the user's personal computer returns to DOS. Unfortunately, IBM uses Alt+Esc as a hot key, which causes a slight amount of confusion when different terminal adapters are used with PS/2s colocated in a "terminal room." At this time, entering the command FT78T initiates the DCA file transfer program for use with an MVS/TSO host environment.

The following illustrates the ease of performing file transfer operations with the software provided with the IRMA terminal emulator adapter board. The file transfer operation illustrated transfers data from the personal computer to the mainframe. Thus, the transfer direction entered was S for send. If data is downloaded from the mainframe to the personal computer, the user enters **R** for receive in response to the `Transfer direction` prompt.

```
C>ft78t
IRMA File Transfer Version 1.25T
Confirm selections prior to transfer (Y/N): y
Transfer direction (R/S): s
Transfer binary file (Y/N): n
Display copy to CON: (Y/N): n
Local filename: ftsample.txt
Data set name: test.data
Operands [none]:
Send to data set 'TEST.DATA' from local file 'FTSAMPLE.TXT'
Ok to continue (Y/N): y

C>
```

When data is sent from the PC to the mainframe, all ASCII characters are translated to EBCDIC. Similarly, when data transfer occurs in the opposite direction, the code conversion is from EBCDIC to ASCII.

IRMAlink

To provide higher speed file transfer capability between members of the IBM PC and PS/2 series and mainframes running under VM/CMS, MVS/SP/TSO, and MVS/XA/TSO operating system environments, Digital Communications Associates developed several high-speed file transfer software programs. Marketed under the name IRMAlink, these programs operate on both the mainframe and personal computer, eliminating the slowness of data transfers through EDIT or XEDIT. In addition, the use of a menu display on the personal computer has simplified file transfer operations so you simply enter the filenames, add the transfer type information, and press a function key.

Figure 14.9 illustrates the completed IRMAlink FT/TSO menu for a 12-function key PC system. After logging onto the mainframe computer and entering the TSO "Ready" mode, you press the IRMA hot key (**Shift+Shift**) to switch back to DOS. Then, entering the command **FTTSO** executes the FT/TSO file transfer program and displays the menu illustrated in Figure 14.9 without the filename, data set name, and transfer type information. After entering those three items, you can press the **F1** key. The PC FTTSO program invokes the IRMAlink program on the mainframe and they operate together to transfer the file.

With IRMA, an IBM PS/2 can emulate several members of the 3270 terminal family, including the 3278 Model 2A (80 × 40 character screen), the 3278 Model 3 (80 × 32 character screen), the 3278 Model 4 (80 × 43 character screen), the 3279 Model 2 (80 × 24 character screen), and the 3279 Model 3 (80 × 32 character screen). Screens on the standard PC series consist of 80 × 25 characters, so IRMA provides a vertical scrolling capability for PS/2 users to view the larger terminal screens it emulates. When emulating the 3279 color display terminal IRMA supports seven colors. In comparison, the IBM PC 3270 Emulation Program Version 3.0 includes a separate sophisticated color select program. This program provides you with the ability to select from 16 colors for protected and unprotected fields and 8 colors for the cursor.

Because IRMA supports multiple logical units (LUs), once PS/2 users log onto their mainframe they can direct printout to their parallel or serial printer as a mainframe addressable IBM 3287 printer while continuing other operations on the mainframe.

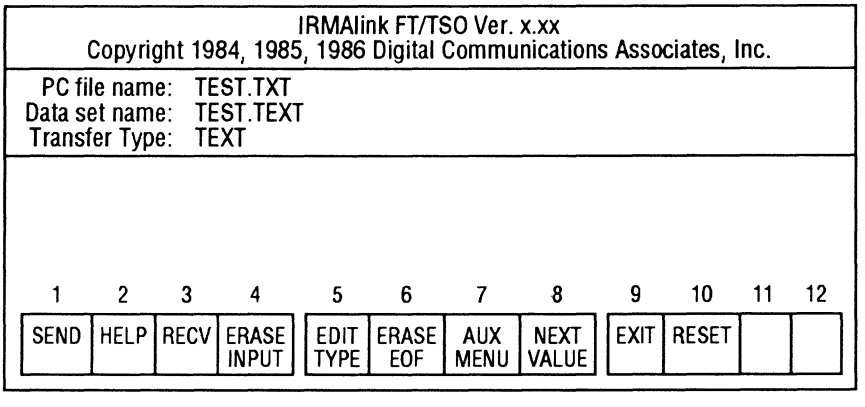


Figure 14.9
IRMAlink FT/TSO File Transfer Menu

Conventional 3278 terminals are not connectable to printers, so the productivity of IRMA users requiring immediate printouts should normally exceed conventional 3270 terminal users who must either wait for the delivery of their print jobs by a member of the organization's production control staff or leave their terminal area and retrieve the print job from the data center or perhaps a remote printer located nearby.

Other valuable IRMA features include the capability to save up to nine screen images to memory for later recall or an infinite number of screens to a previously named file. Then, screens saved to a file or files can be retrieved and displayed or printed.

CXI and Other Products

The 3270-PC Connection from CXI, Inc., is an add-in board and software that is installed in a personal computer to provide micro-to-mainframe communications by emulating an IBM 3270 PC. This product permits up to five interactive host applications to be windowed as well as one PC-DOS session and two notepads. Similar to IRMA, with the 3270-PC Connection you can use one or more PC attached parallel or serial printers as mainframe addressable IBM 3287 printers. A similar product to CXI's 3270-PC Connection is Forte Data Systems' 3270/PC package, which enables a PC bus personal computer to emulate a 3270 PC.

To help you compare different emulator adapter board products, Table 14.5 contains a list of selection features to consider as you purchase a board. Although most entries are self-explanatory, three deserve additional elaboration.

The user-definable keyboard refers to the capability of the emulator to permit the keys or key sequences on an IBM PS/2 keyboard to be remapped. Although this may

Table 14.5
3270 Emulation Adapter Selection Features

Feature	Your Requirement
Emulator Type	Terminal Emulated Controller Emulated
Number of Host Sessions Supported	
Host Printer Sessions Support	
File Transfer Support	TSO EDIT CMS EDIT CICS Other
User-definable keyboard	
Interface Data Rate	
Keyboard Template Available	
PC Memory Requirement	
Adapter board size and type	
Hot Key	
Screen image saving to files	
Screen image saving to memory	

appear trivial at first, for installations using a number of different products to access their 3270 network this capability can reduce potential confusion of operations. As an example, some emulator products such as DCA's IRMA use the ALT+digit combination for PF1 through PF10 keys. Other emulators, such as IBM products, use the function keys for the first 10 or 12 PF keys. Thus, the ability to redefine the keyboard can enable an organization to use different vendor products in a common manner.

The adapter board size and type reference the physical length of the board and the bus it is designed to support. Some half-size boards can be installed in the short slot in a PS/2 Model 25, whereas a standard length board requires insertion into a fully available system expansion slot in that computer.

The number of host sessions supported refers to whether or not the emulator adapter can emulate a distributed-function terminal (DFT). The IBM 3278 and 3279 terminals are single host session display devices. In comparison, new IBM terminals include a DFT mode that enables up to four mainframe sessions to be executed from the remote terminal. Some adapter boards emulate a single session terminal, whereas other adapters emulate a DFT terminal, enabling multiple sessions to be controlled from a single PC.

Extending the Connection

If personal computers are located at a distance from a control unit or a communications controller, coaxial cable cannot be used to connect the PC to the other device. Although you could use a protocol converter functioning as a control unit emulator and connect a PC via a dedicated communications link to the control unit, in certain situations, PC activity in accessing the mainframe does not warrant the expenses associated with a dedicated circuit. Similarly, although you can employ a leased line to connect a personal computer to a communications controller, in many instances equipment that permits the PC to use the direct distance dial network may be more economically feasible than requiring the use of leased lines.

Figure 14.10 illustrates how personal computers can be interfaced to a control unit and a communications controller via the direct distance dial network or via the installation of a leased line between the PC and the device it is to be connected to, with the method employed normally governed by the anticipated activity of the PC. Thus, PCs with a low connect time would most likely be connected through the use of the direct distance dial network. At some point in time, the cost of numerous dialed calls during the month would equal the monthly cost of a leased line. At this point, it would be more economical to obtain a leased line to connect the PC to the communications controller or to the control unit.

For both types of communications facilities, a protocol converter is normally installed at the controller or control unit site and connected via coaxial cable to that device. These protocol converters have a serial RS-232-C interface, enabling them to be connected via a leased line or via the switched telephone network to an asynchronous ASCII device, such as a standard IBM PS/2. Because the protocol converter may not produce an ASCII code output compatible with an IBM PC, some third-party vendors offer a software program that provides the personal computer with asynchronous terminal emulation capability that, in turn, is compatible with the asynchronous terminal format supported by the protocol converter.

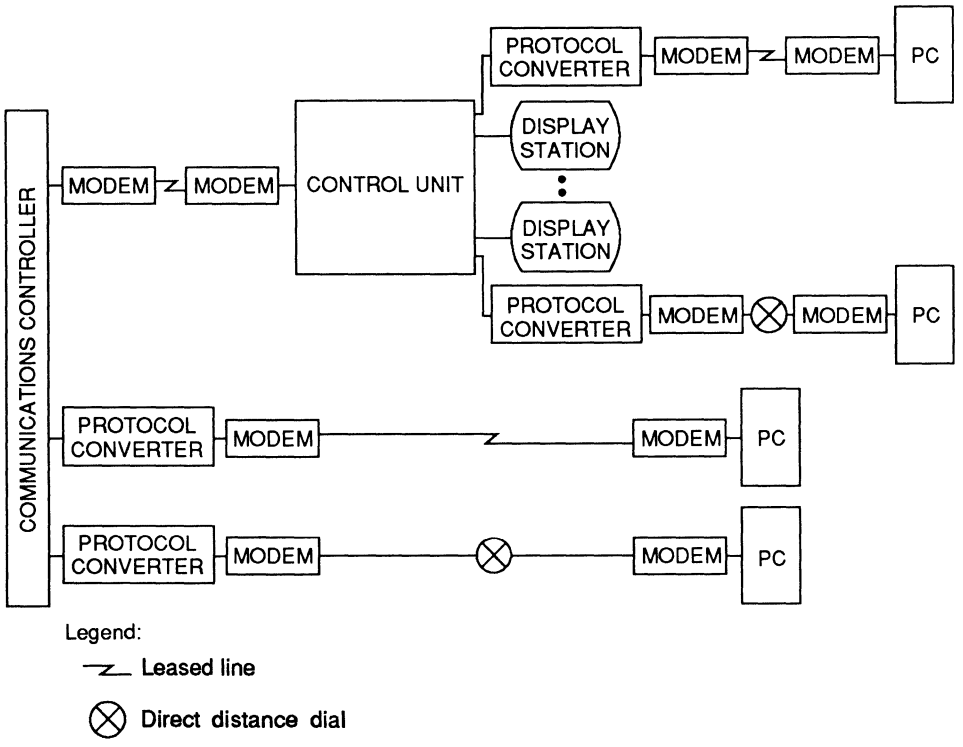


Figure 14.10
Remotely Connecting to Controllers

Remote Job Entry Communications

In addition to making a personal computer appear to the mainframe as an interactive 3278 type terminal, there are other products that convert the PC into a remote job entry (RJE) workstation. Products in this category for the personal computer emulate a 2780, 3780, 2770, 3770, or 3741 workstation and permit the PC user to communicate with the mainframe as a batch terminal. Two examples of batch terminal emulators available for use with IBM PCs are the AST-3780 from AST Research and the IRMAcom/3770 from DCA.

The AST-3780 is a hardware and software package that can be employed to enable an IBM PC bus computer to emulate common bisynchronous RJE workstations to include 2780, 3780, 2770, and 3741 devices. This package consists of an adapter board that is inserted into an expansion slot in the system unit of a PC bus computer and appropriate software that allows the PC to communicate as a bisynchronous workstation to IBM, Honeywell, and other mainframes that support such workstations. Included in the software are printer forms control functions that enable a printer attached to the personal computer to function as a workstation printer. Other software features included enable dynamic device selection between displays, printer, and disk as well as the ability to perform text compression, which is a feature of the 3780 protocol. In addition to connecting a PC to a mainframe, two PCs equipped with the AST-3780 package can

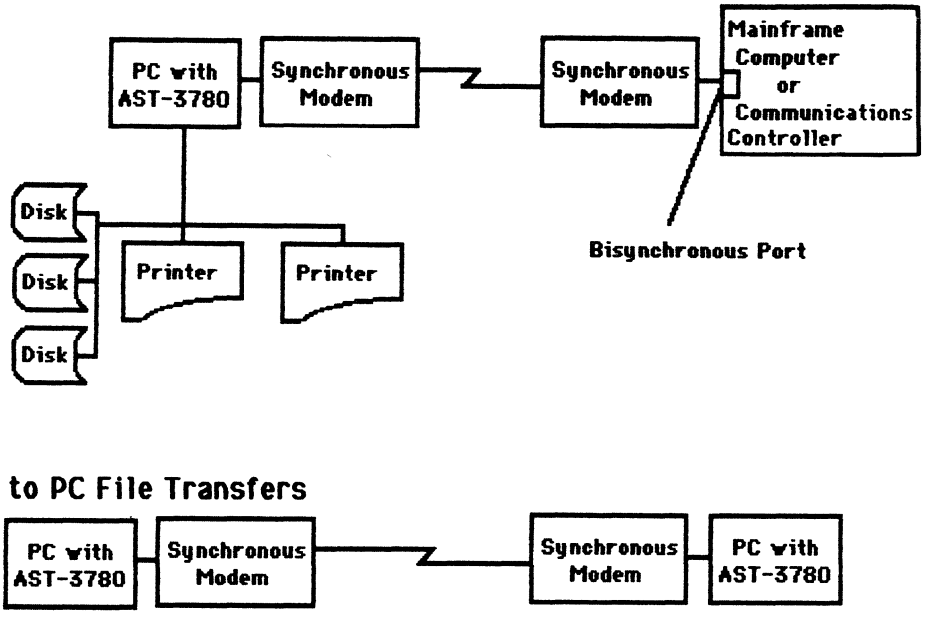
communicate with one another. When connected to a mainframe, the AST-3780 package enables your PS/2 to interact with such mainframe packages as HASP, JES1, POWER, and RES. Figure 14.11 illustrates two typical PC equipped AST-3780 networking applications. In the top part of Figure 14.11, a PC equipped with the AST-3780 Communications Product functions as a remote batch workstation, connected via synchronous modems to a bisynchronous port on a mainframe computer or communications controller attached to a mainframe. In the lower part of Figure 14.11, two PCs, each with an AST-3780 Communications Product, communicate with one another similar to the manner in which two conventional workstations would communicate with one another.

Hardware Alternatives

Micro/Host Software

Because both mainframe computers and PCs are programmable devices, hardware boxes and add-on boards may be eliminated in certain situations using an appropriate communications software package. One such package being marketed is SIM3278 from SIMWARE of Nepean, Ontario. SIM3278/VM is a program package that can be installed on an IBM virtual mainframe (VM) and provides access to such VM applications as DMS, SPF, PROFS, XEDIT, and STAIRS, as well as to such operating systems as OS/VS1, MVS, and DOS/VSE running under CP. This program permits any ASCII terminal supported by SIM3278—including IBM PCs and PS/2s—to be connected to the mainframe computer via a direct cable, a leased line, or the dial-up telephone

Figure 14.11
Using the AST-3780



network. Once connected to a VM system, your PS/2 can connect to SIM3278/VM, and the Control Program (CP) initializes a logical 3270 screen that is associated with the terminal.

Some of the key features of SIM3278 are its multiple session manager, on-line help facility, and split-screen window management facility. The product's multiple session manager accepts log-ons of up to 12 "userids" at one time, and a simple command toggles a user from one session to another. As an example, users can easily switch from a CMS session to TSO or to another CMS session and so on, all without having to log off or disconnect from the mainframe. Because the PC's keyboard differs from that of a 3278 terminal, it is often difficult to remember the PC keystrokes required to emulate a 3278 terminal function. To alleviate this problem, a user of SIM3278 can type **#HELP** at any time to view a screen illustrating the PC keystrokes required to emulate 3278 functions. The vendor's split-screen, window management facility can display up to four concurrent sessions, providing a large portion of the power of a 3270 PC to a conventional PS/2. To obtain this capability you must obtain the vendor's SIM3278/PC software for use on a PS/2 that is designed to operate in conjunction with the vendor's SIM3278 program for the mainframe computer. Together, the two programs provide complete 3270 terminal emulation as well as file transfer capability and printer support, representing a novel approach to linking PCs into a 3270 environment since it alleviates the procurement of specialized hardware.

Packet Network Facilities

Due to the extensive number of companies that have 3270 type networks, attaching PS/2s to those networks has become an all too familiar problem. Whereas the previously discussed methods have been gainfully employed by many organizations, in certain instances, those methods may not be economically viable to implement. For an example of a situation where the previously discussed 3270 interface strategies might be uneconomical, consider the following hypothetical situation.

Suppose an organization's mainframe computer is located in New York City and an existing 3270 network consists of leased lines from New York to regional offices located in Boston, Chicago, Dallas, Denver, and Los Angeles. At each of the five regional offices, further assume that a 3174 or 3274 control unit is installed that services a number of conventional 3278/9 terminals located in each office. After the 3270 network was established, a large number of small area offices of the company obtained personal computers to perform local processing of data. In addition to their initial use, suppose each area office manager had a requirement for his or her personal computer to access a 3270 application program on the corporate mainframe. After studying the anticipated activity of the area offices, the corporate communications manager determined that each area office would only require access to the mainframe for approximately 1 hour per week. If the organization has 80 area offices and most of them are located at least 100 miles from a regional office, how can the personal computers be economically connected to the mainframe?

Based on the preceding information, it would be very expensive to obtain hardware for each personal computer to make it function as a 3270 type terminal. In addition, although the 1 hour per week connect time is relatively low, when multiplied by 80

computers, the cost of long distance transmission over the switched telephone network would be considerable. Thus, from an economic perspective, the previously discussed hardware solutions and the micro/host software method of integrating personal computers into a 3270 network might not be economically practical. To satisfy organizations with networking problems similar to this hypothetical example, several value-added carrier packet networks have implemented what is known as 3270 access.

Under 3270 access, value-added carriers have implemented protocol conversion into the *packet assembly/disassembly (PAD)* computers they install at a company's mainframe location, as illustrated in Figure 14.12. Under the 3270 access concept, both personal computers and conventional asynchronous terminals dial a packet network node and are then routed through the packet network to the PAD, where the asynchronous data is converted into a synchronous, 3270 format. Because the packet network's protocol conversion software would have to be extensive to support the myriad types of personal computers produced by various manufacturers, most packet networks support one or at most only a few asynchronous protocols that are then converted to emulate a 3270 type terminal by the PAD. As an example of this, Tymnet will only permit VT-100 operating terminals and personal computers using a VT-100 emulator to be converted into a 3270 protocol. This means that PC users must obtain a VT-100 emulator program for use on their PC if they desire to use Tymnet's 3270 access facility.

A significant extension to the 3270 access concept occurred in late 1986 with the introduction of X.25 communications adapter boards for use in the IBM PC and compatible computers. Marketed by Western Digital and Eicon Technology Corporation, an X.25 board and support software enables a personal computer to directly access a mainframe computer via a packet network, functioning as a PAD in converting the

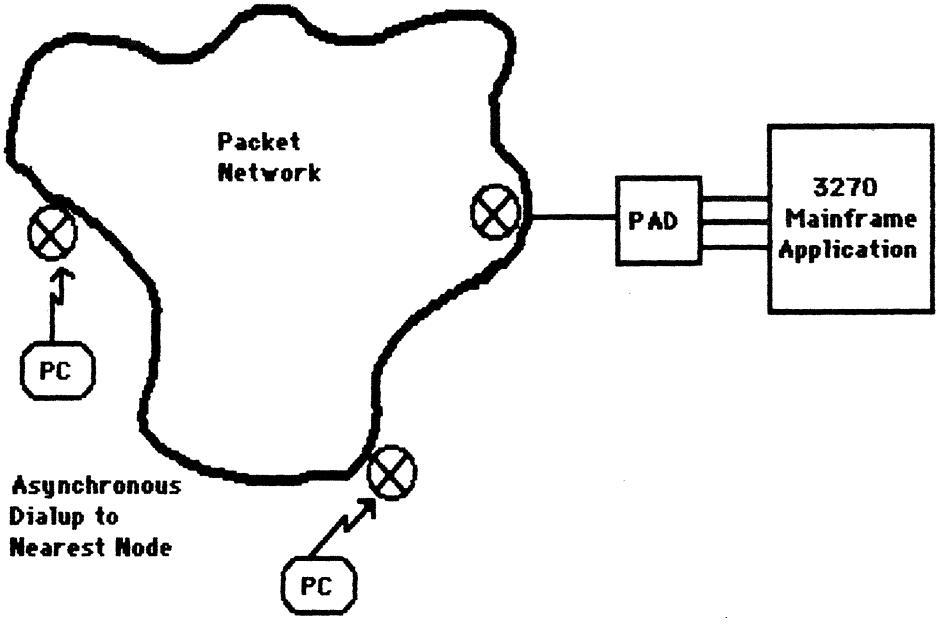


Figure 14.12
Packet Network 3270 Access

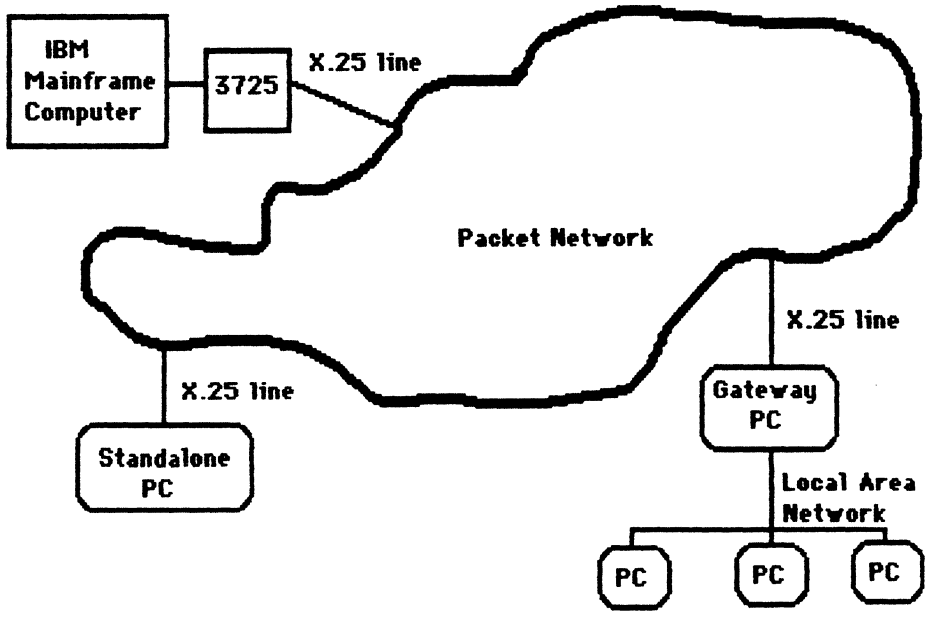


Figure 14.13
Expanding 3270 Access Methods

PS/2's asynchronous ASCII data into a synchronous X.25 protocol. As illustrated in the lower left portion of Figure 14.13, PCs equipped with an X.25 adapter board can communicate with a mainframe, such as an IBM 33XX or 43XX series computer, without the use of a separate PAD. In this instance it is assumed that an IBM 3725 communications controller supports the X.25 protocol and has one or more X.25 lines connected to the packet network. Taking X.25 access one step further, in 1988 several vendors introduced combined X.25 adapter boards and internal modems, whereas other vendors introduced standalone modems that include a built-in PAD capability.

Taking X.25 technology further, Eicon Technology Corporation developed an adapter board and software that can be used in a PC to obtain a local area network gateway to a remotely located mainframe via a packet network. This concept is illustrated in the lower right portion of Figure 14.13.

When an X.25 adapter is used in a personal computer, the packet network is no longer required to perform protocol conversion. Thus, although users still incur a data transportation charge based on session duration and the quantity of characters or packets transported, the monthly cost associated with the use of a PAD is eliminated.

15 / Local Area Networks

Although local area networks (LANs) are probably discussed more frequently in trade magazines, seminars, and exhibitions than most other communications topics, until recently their level of use was far below their level of discussion. Now, this is rapidly changing due to the tremendous growth in the use of personal computers in corporations, service organizations, and on college campuses.

One of the limiting factors of personal computers is their use primarily as isolated workstations. This means that in most organizations it is difficult for personal computer users to share data and peripheral devices, because such sharing normally requires physical activity. There are direct economical benefits of sharing data and peripheral devices, as well as productivity gains resulting from the integration of personal computers into a common network. These factors have contributed to an increasing demand for local area network products, which is the focus of this chapter.

This chapter first describes the major benefits derived from the use of local area networks and their relationship to typical network applications. Next, a section looks at the major areas of local area network technology and the effects these areas have on the efficiency and operational capability of such networks. Here, the examination focuses on network topology, transmission media, and the major access methods employed in LANs. Using the previous material as a base, you then get a brief look at what is known as the IEEE 802 series of LAN standards. The chapter concludes by examining several local area networks designed for use by personal computer systems.

Utilization Benefits

In its simplest form, a local area network can be considered as a cable that provides an electronic highway for the transportation of information to and from different devices connected to the network. By providing the capability to route data between different devices connected to a common network, numerous benefits may accrue to individual personal computer users that may not be available to single user systems. Such benefits can include sharing peripherals, common access to data files and programs, equipment compatibility, dispersion of equipment, and an increase in the probability of access to data.

Peripheral Sharing

Even the largest organization cannot afford to purchase fully configured personal computer systems, including large capacity disk drives, laser printers, plotters, and other

peripheral devices whose use may only require a small portion of the time a PS/2 system is in operation. By linking personal computers into a local area network, each PS/2 becomes a workstation on the net, obtaining the capability to share access to other devices connected to the network. Thus, LAN users can access resources that would probably be too expensive to justify for individual computer systems.

Common Access

Normally, most software sold for use on personal computers contains a license agreement known as a "shrink wrapped" license, due to its inclusion in the cellophane-covered software package. The typical terms of such licenses state that the software is licensed for use on one computer system. This means that an office with 50 personal computers would theoretically require 50 spreadsheet programs, 50 database management programs, and so on, if each PC user required access to a common set of programs. Because of the typical terms in a software license agreement, it would be illegal to place one version of a program on the network for all of the PS/2 users to access.

However, recognizing the proliferation of local area networks, many software vendors have implemented multiuser versions of their products that are designed for operation on local area networks. In addition to being less costly than obtaining multiple copies of individual programs, these programs are normally rewritten to support concurrent operation. This means, as an example, that one user concurrently using a database program to access a common file already being modified by a second user would be temporarily "locked out" of a record or field within a record until the second user completed his or her activity. Thus, in addition to providing common access to programs, a local area network may permit users to share data files; the network must employ means to ensure the integrity of data in shared files, as well.

Equipment Compatibility

Unfortunately, a LAN cannot take a program written for one manufacturer's personal computer and convert it to operate on a personal computer manufactured by a different vendor when the two devices are incompatible. Thus, even though one might be able to transfer a BASIC language program written to operate on an IBM PS/2 through a local area network to an Apple Macintosh computer connected to that network, the probability of the program operating on the Apple personal computer without modification would be minimal. What the LAN does provide is a common interface between different types of equipment produced by the same manufacturer or by different manufacturers, which permits users to share common peripherals (storage devices, printers, and so on). Thus, the LAN can free the organization to a degree from dependence on any single vendor, as well as provide a mechanism to integrate the use of peripheral devices that can be shared by many users. In addition, certain software products may enable a larger degree of equipment compatibility if they generate a standardized format data file. In such situations, an Apple user could then use a local area network to access a spreadsheet model previously created on an IBM PS/2. Assuming an Apple Macintosh version of the spreadsheet program is available, the Apple user could then use that version of the program with the IBM PS/2 model previously saved to a common storage area of the network.

Distribution of Hardware

Prior to the advent of personal computers, most large organizations centralized the physical location of mainframe terminals in "terminal rooms." Typically, such terminal rooms contained a number of cubicles where a terminal and printer were located on a desk. Members of the organization would either contend for access to a terminal or place their names on a chart to reserve a particular period of time.

In the early 1980s, the initial entry of personal computers in organizations was often financially disguised as the purchase of office equipment, calculators, and the like by office managers attempting to bypass the firm's data processing department. These desktop micros were usually placed in a convenient physical location near the user. As the use of personal computers proliferated, many organizations implemented policies concerning the acquisition and use of PCs. Because some users only required a personal computer to develop and fine-tune spreadsheet models, whereas other users desired to prepare formal reports or access information utilities, the cost of providing an individually tailored hardware system for each user became prohibitive. This caused many organizations to revert to the "terminal room" concept, locating groups of personal computers in a confined area.

Through a local area network, it becomes economically feasible to distribute personal computers to the most convenient physical locations within a facility while retaining the shared use of expensive peripheral devices. Thus, the local area network can promote the placement of personal computers near the source of the users' physical workspaces or application areas.

Reliability of Data Access

One of the major advantages of implementing a local area network is the distributed processing environment it provides to users. This means users may be able to access applications independent of the operating status of any particular personal computer. In addition, if applications are located on the network, the failure of a mainframe computer will not lock out the user from access to his or her application.

Common Gateway

When one personal computer is used as a common gateway to a mainframe computer, other PCs on the network can obtain access through the gateway to the mainframe. Through the use of a gateway, a number of economical communications networking strategies can be considered to connect LANs to remotely located or colocated mainframes. As an example, one personal computer on the LAN could be connected via a leased line to a remotely located mainframe. If the transmission requirements of the personal computers on the LAN for access to the mainframe do not justify the use of leased lines, alternative access methods—including asynchronous dial-up to the mainframe via the PSTN or a packet network—could be considered. In addition, if the communications controller connected to the mainframe supports X.25, the installation of an X.25 adapter board in the gateway personal computer enables users to access the mainframe via a packet network without the use of a PAD.

LAN Disadvantages

Although the benefits LANs can provide are considerable, as with most technologies there are certain problems and costs that must be considered before you implement the technology. Among the potential problem areas associated with LANs are cabling, software support, resources required to implement a LAN, and the incompatibility of many hardware and software products.

LANs that use coaxial cable can be very expensive to install, especially if a new conduit must be constructed if an existing conduit is filled to its maximum capacity. Concerning software support, most local area networks contain hardware and software products from many vendors. This can mean that you have to contact multiple vendors for network hardware, network software, and the application software when an application is not working correctly.

The resources required to implement a local area network can be considerable with respect to both economics and effort. In addition to the normal system acquisition process in which you evaluate and select a system based on a prior development of system requirements, most LANs with 20 or more stations require a system administrator to control and coordinate usage. Finally, due to the rapid evolution of LAN products, it is quite common for certain hardware products to be incompatible with some LAN software products marketed by the same vendor. In addition, it is also very common for many software products from a LAN vendor or other vendors to be incompatible with each other. Although incompatibility within a vendor's product line is normally the exception rather than the rule, it becomes far more pronounced when you use LAN software, because such software typically resides in RAM above 640K, causing memory conflicts with utilities, pop-up programs, and other applications. Due to this, existing and potential LAN users must carefully evaluate products with respect to their compatibility.

Technological Characteristics

Although a local area network by nature is a limited distance transmission system, the variety of vendor product offerings is anything but limited. Products currently marketed as well as expected offerings run the gamut with respect to their functionality, operation, application support, and hardware and software requirements. This section examines the technological characteristics of local area networks, including their topologies, transmission media, the access method used to transmit data on the network, and the hardware and software required to make the network operate. In addition, because communications connectivity is offered by the IBM Cabling System, this section examines its features after discussing the different types of media used in local area networks.

Topology

The *topology* of a local area network refers to the structure or geometric layout of the cable used to interconnect stations on the net. Unlike conventional data communications networks that can be configured in a variety of ways by the addition of hardware and software, most local area networks are designed to operate based on the interconnection

of stations that follow a specific topology. The most common topologies used in LANs include the bus, the ring, and the star, as illustrated in Figure 15.1.

In a bus topology structure, a cable is usually laid out as one long branch onto which each station on the net is interconnected to the main data highway. Although this type of structure permits any station on the net to talk to another station, rules are required to govern the action necessary to recover from such situations as when two stations attempt to communicate at the same time. Later, this chapter examines the relationship between the network topology, the method employed to access the network, and the transmission medium employed in building the network.

In a ring topology, a single cable that forms the main data highway is shaped into a ring. Similar to the bus topology, branches interconnect stations to one another via the ring. Thus, a ring topology can be considered to be a looped bus. Typically, the access method employed in a ring topology requires data to circulate around the ring, with a special set of rules governing when each station connection to the network can transmit data.

The third major local area network topology is the star structure, illustrated in the lowest portion of Figure 15.1. In a star network, each station on the net is connected to a network controller. Then, access from any one station on the net to another station can be accomplished through the network controller. Here the network controller can

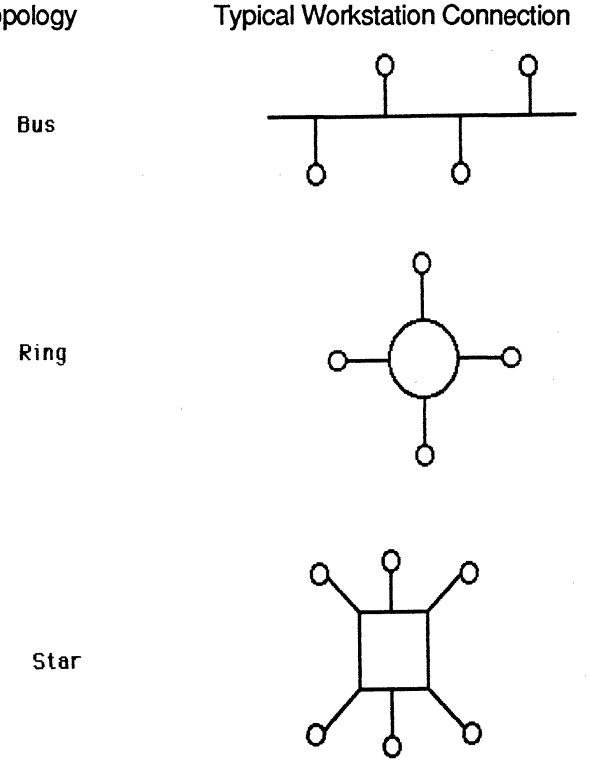


Figure 15.1
LAN Topology

be viewed as functioning similar to a telephone switchboard, because access from one station to another station on the net can only occur through the central device.

Some networks are a mixture of topologies. As an example, the IBM Token Ring Network can actually be considered to be a "Star Ring" topology, because up to eight personal computers are first connected to a common device known as a *Multistation Access Unit (MAU)*, which in turn is connected in a ring topology to other MAUs. Later this chapter describes the IBM Token Ring Network in detail.

Although there is a close relationship among the topology of the network, its transmission media, and the method used to access the net, you can study topology as a separate entity and make several generalized observations. First, in a star network the failure of the network controller renders the entire network inoperative. This results from the fact that all data flow on the network must pass through the network controller. On the positive side, the star topology normally exists within most buildings in the form of telephone wires that are routed to a switchboard. This means that a local area network that can use in-place twisted-pair telephone wires is normally simple to implement and usually very economical.

In a ring network, the failure of any node connected to the ring usually inhibits data flow around the ring. Because data travels in a circular path on a ring network, any cable break has the same effect as the failure of the network controller in a star structured network. Each network station is connected to the next network station, so it is usually easier to install the cable for a ring network. In comparison, if existing telephone wires are not available you have to cable each station in a star network to the network controller, which can result in the installation of very long cable runs.

In a bus-structured network, data is normally transmitted from one station to all stations located on the network, with the destination address appended to each transmitted data block. As part of the access protocol only the station with the destination address in the transmitted data block responds to the data. This transmission concept means that a break in the bus may only affect network stations on one side of the break that wish to communicate with stations on the other side of the break. Thus, unless a network station functioning as the primary network storage device becomes inoperative, a failure in a bus structured network is usually less serious than if a failure occurs in a ring network.

Transmission Medium

The transmission medium used in a local area network can range in scope from twisted-pair wire, such as is used in conventional telephone lines, to coaxial cable, fiber optic cable, and the atmosphere (which is used by some esoteric transmission schemes, including FM radio). Each transmission medium has a number of advantages and disadvantages associated with its use. The primary differences between media concern their cost and ease of installation; the bandwidth of the cable, which may permit only one or several transmission sessions to occur simultaneously; the maximum speed of communications permitted; and the geographic scope of the network that the medium supports.

Twisted-Pair Wire

In addition to being inexpensive, twisted-pair wire is very easy to install. Normally, a screwdriver and perhaps a pocket knife are the only tools required for its installation.

Anyone who has hooked up a pair of speakers to a stereo set normally has the ability to install this transmission medium. Although inexpensive and easy to install, twisted-pair wire is very susceptible to noise generated by fluorescent light ballasts and electrical machinery. This noise can affect the error rate of data transmitted on the network, although lead-shielded twisted-pair cable can be employed, which provides a high degree of immunity to line noise.

Because the bandwidth of twisted-pair cable is considerably less than coaxial or fiber optic cable, normally only one signal is transmitted on this cable at any point in time. This signaling technique is known as *baseband signaling* and should be compared to the broadband signaling capability of coaxial and fiber optic cable. Broadband signaling permits a cable to be subdivided by its frequency bandwidth into many individual subchannels, with each subchannel permitting an independent communications session to occur simultaneously with other sessions transpiring on other subchannels. Figure 15.2 illustrates the difference between baseband and broadband signaling. Note that although a twisted-pair wire system can be used to transmit both voice and data, the data transmission is baseband, because only one channel is normally used for data. In comparison, a broadband system on coaxial cable can be designed to carry voice and several subchannels of data as well as facsimile and video transmissions. Another constraint of twisted-pair wire is the rate at which data can flow on the net and the distance it can flow. Although data rates up to 10 megabits (Mbps) can be achieved, normally local area networks employing twisted-pair wiring operate at a lower data rate. In addition, twisted-pair systems normally cover a limited distance measured in terms of thousands of feet, whereas a coaxial based system may be limited in terms of miles.

Coaxial Cable

Coaxial cable consists of a center conductor copper wire that is covered by an insulator known as a *dielectric.* An overlapping woven copper mesh surrounds the dielectric and the mesh, in turn, is covered by a protective jacket that can consist of polyethylene or aluminum. Figure 15.3 illustrates the composition of a typical coaxial cable; note, however, that over 100 types of coaxial cable are currently marketed. The key differences between such cables involve the number of conductors contained in the cable; the die-

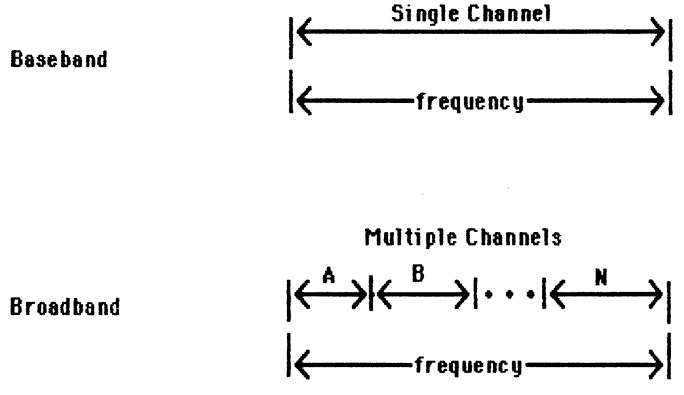


Figure 15.2
Baseband Versus Broadband Signaling

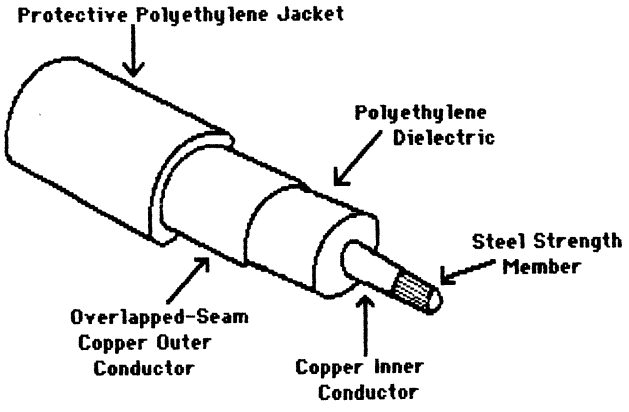


Figure 15.3
Coaxial Cable

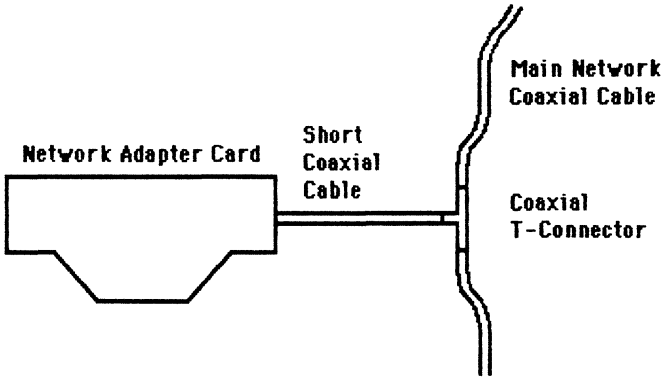


Figure 15.4
Hardware Interface to Coaxial Cable

lectric employed; and the type of protective jacket and material used to provide strength to the cable, which allows it to be pulled through conduits without breaking.

Hardware Interface

A coaxial cable with a polyethylene jacket is normally used for baseband signaling. Data is transmitted from stations on the network to the baseband cable in a digital format and the connection from each station to the cable uses a simple coaxial T-connector. Figure 15.4 illustrates the hardware interface to a coaxial cable of a typical baseband local area network. Here the network adapter card is a hardware device that contains the logic to control network access and is inserted in one of the expansion slots in the system unit of an IBM PS/2. At the rear of the computer's system unit, a short section of coaxial cable is used to connect the network adapter card to the baseband cable via a T-connector.

Because data on a baseband network travels in a digital form, those signals can be easily regenerated by the use of a device known as a *line driver* or *data regenerator.* The line driver or data regenerator is a low-cost device constructed to look for a pulse rise; on detecting the occurrence of the rise, it disregards the entire pulse and regen-

erates an entirely new pulse. Thus, one can install low-cost line drives into a baseband coaxial network to extend the distance transmission can occur on the cable. Typically, a coaxial cable baseband system can cover an area of several miles and may contain hundreds to thousands of stations on the network.

Broadband Coaxial Cable

To obtain independent subchannels derived by frequency on coaxial cable broadband transmission requires a method to translate the digital signals from PCs and other workstations into appropriate frequencies. This translation process uses radio frequency (RF) modems that modulate the digital data into analog signals and convert or demodulate received analog signals into digital signals. Because signals are transmitted at one frequency and received at a different frequency, a "head end" or frequency translator is also required for broadband transmission on coaxial cable. This device is also known as a *remodulator*, because it simply converts the signals from one subchannel to another subchannel.

The requirement for modems and frequency translators normally makes broadband transmission more expensive than baseband. Although the ability of broadband to support multiple channels provides an aggregate data transmission capacity that exceeds baseband, in general, baseband transmission permits a higher per channel data flow. This is an important consideration for mainframe-to-mainframe communications when massive amounts of data must be moved, but for most personal computer file transfer operations the speed of either baseband or broadband transmission should be sufficient. This fact may be better understood by comparing the typical transmission rates obtainable on baseband and broadband networks to drive a high-speed dot-matrix printer and the differences between the time required to transmit data on the network and the time required to print the data.

Typical transmission speeds on baseband and broadband networks range from 2 to 10 Mbps. In comparison, a high-speed dot-matrix printer operating at 120 cps requires approximately 200 seconds to print 1 second's worth of data transmitted at 2 Mbps and 1000 seconds to print 1 second's worth of data transmitted at 10 Mbps.

Fiber Optic Cable

Fiber optic cable is a transmission medium for light energy and as such provides a very high bandwidth, permitting data rates ranging up to billions of bits per second. The fiber optic cable consists of a thin core of glass or plastic surrounded by a protective shield. Several shielded fibers in turn are bundled in a jacket with a central member of aluminum or steel employed for tensile strength.

Digital data represented by electrical energy must be converted into light energy for transmission on a fiber optic cable. This is normally accomplished by a low-power laser or through the use of a light emitting diode and appropriate circuitry. At the receiver, light energy must be reconverted into electrical energy. Normally, a device known as a *photo detector*—as well as appropriate circuitry to regenerate the digital pulses and an amplifier—convert the received light energy into its original digital format.

In addition to the high bandwidth of fiber optic cables, they offer users several additional advantages in comparison to conventional transmission media. Because data

travels in the form of light, it is immune to electrical interference, and building codes that may require expensive conduits to be installed for conventional cables are usually unnecessary. Similarly, fiber optic cable can be installed through areas where the flow of electricity could be dangerous, because only light flows through the optic cables.

Because each fiber provides just a single, unidirectional transmission path, a minimum of two cables is required to connect all transmitters to all receivers on a network built using fiber optic cable. Due to the higher cost of fiber optic cable than coaxial or twisted-pair, the dual cable requirement of fiber optic cables makes them very expensive. In addition, it is very difficult to splice such cable, which usually involves sophisticated equipment and skilled installers to implement. Similarly, once this type of network is installed, it is difficult to modify the network. Currently, the cost of the cable and the difficulty of installation and modification make the use of fiber optic based local area networks impractical for many commercial applications. With the cost of the fiber optic cable declining and improvements expected to simplify the installation and modification of networks using this type of cable, the next few years may witness a profound movement toward adoption of this transmission medium.

IBM Cabling System

The IBM Cabling System was introduced in 1984 as a mechanism to support the networking requirements of office environments. By defining standards for cables, connectors, faceplates, distribution panels, and other facilities, IBM's Cabling System is designed to support the interconnection of personal computers, conventional terminals, mainframe computers, and office systems. In addition, this system permits devices to be moved from one location to another or added to a network through a simple connection to the Cabling System's wall plates or surface mounts.

Cable Types

The IBM Cabling System specifies seven different cabling categories. Depending on the type of cable selected, you can install the selected wiring indoors, outdoors, under a carpet, or in ducts and other air spaces.

The IBM Cabling System uses wire that conforms to the *American wire gauge (AWG).* AWG is a unit of measurement for the wire diameter. As the wire diameter gets larger, the AWG number decreases, in effect creating an inverse relationship between wire diameter and AWG. The IBM Cabling System uses wire between 22 AWG (.644 mm) and 26 AWG (.405 mm). Because a larger diameter wire has less resistance to current flow than has a smaller wire diameter, a smaller AWG permits cabling distances to be extended in comparison to a higher AWG cable.

Type 1

The IBM Cabling System Type 1 cable contains two twisted pairs of 22 AWG conductors. Each pair is shielded with a foil wrapping, and both pairs are surrounded by an outer braided shield. Type 1 cable provides the largest cabling distance and is available in two designs—plenum and nonplenum. *Plenum cable* can be installed without the use of a conduit, whereas *nonplenum cable* requires a conduit.

Type 2

Type 2 cable is actually Type 1 cable with the addition of four pairs of 22 AWG conductors for telephone usage. Due to this, Type 1 cable is also referred to as *data-grade twisted pair cable,* whereas Type 2 cable is known as *two data-grade* and *four voice-grade twisted pair.* Due to its voice capability, Type 2 cable can support PBX interconnections. Like Type 1 cable, Type 2 cable supports plenum and nonplenum designs.

Type 3

Type 3 cable is conventional twisted pair, telephone wire. Both 22 AWG and 24 AWG conductors are supported by this cable type. One common use of Type 3 cable is to connect PS/2s to MAUs in a Token Ring Network.

Type 5

Type 5 cable is fiber optic cable. Two 100/140 micron optical fibers are contained in a Type 5 cable. This cable is suitable for indoor, nonplenum installation or outdoor aerial installation. Due to the extended transmission distance obtainable with fiber optic cable, Type 5 cable is used in conjunction with the IBM 8219 Token Ring Network Optical Fiber Repeater to interconnect two MAUs up to 6600 feet (2 kilometers) from one another.

Type 6

Type 6 cable contains two twisted pairs of 26 AWG conductors for data communications. It is available for nonplenum applications only, and its smaller diameter than Type 1 cable makes it slightly more flexible.

Type 8

Type 8 cable is designed for installation under a carpet. This cable contains two individually shielded, parallel pairs of 26 AWG conductors.

Type 9

Type 9 cable is essentially a low-cost version of Type 1 cable. Like Type 1, Type 9 cable consists of two twisted pairs of data cable. However, 26 AWG conductors are used in place of the 22 AWG wire used in Type 1 cable. As a result of the use of a smaller diameter cable, transmission distances on Type 9 cable are approximately two-thirds that obtainable using Type 1 cable.

Connectors

The IBM Cabling System includes connectors for terminating both data and voice conductors. The data connector has a unique design based on the development of a latching mechanism that permits it to mate with another, identical connector.

Figure 15.5 illustrates the IBM Cabling System data connector. Its design makes it self-shorting when it is disconnected from another connector. This provides a Token Ring network with electrical continuity when a station is disconnected. Unfortunately, the data connector is very expensive compared to an RS-232 connector with the typical

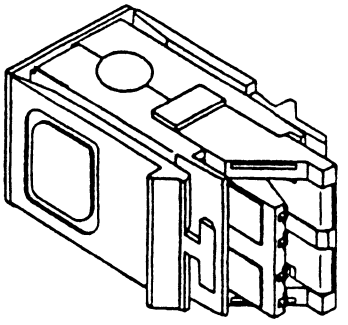


Figure 15.5
Cabling System Data Connector

retail price of the data connector between $8 and $10, whereas RS-232 connectors cost approximately $3.

Due to the high cost of data connectors and cable, the acceptance of the IBM Cabling System by users has been slow. Because it provides a standard system of office interconnectivity, its usage may significantly increase if it becomes more economical in comparison to the cost of conventional connectors and cable.

Access Method

If the topology of a local area network can be compared to a data highway, then the access method might be viewed as the set of rules that enable data from one workstation to successfully reach its destination using the data highway. Without such rules, it is quite possible for two messages sent to the same or a different address by two different workstations to collide, with the result that neither message reaches its destination.

Prior to discussing how access methods work, first examine the two basic types of devices that can be attached to a local area network so you will have an appreciation for the work the access method must accomplish.

Listeners and Talkers

The operating mode of each device can be categorized as being a "listener" or a "talker." Some devices, like printers, only receive data and thus only operate as listeners. Other devices, such as a personal computer, can either transmit or receive data and are capable of operating in both modes. In a baseband signaling environment where only one channel exists or on an individual channel on a broadband system, if several talkers wish to communicate at the same time a collision occurs unless they use a scheme that defines when each device can talk and, in the event of a collision, what events must transpire to avoid its recurrence.

For data to correctly reach its destination, each listener must have a unique address, and its network equipment must be designed to respond to a message on the net only when it recognizes its address. Thus, the primary goals in the design of an access method are to minimize the potential for data collision and provide a mechanism for

corrective action when data collides, as well as to ensure that an addressing scheme is employed to enable messages to reach their destination.

The three access methods primarily employed in PC based local area networks are Carrier-Sense Multiple Access/Collision Detection (CSMA/CD), Carrier-Sense Multiple Access/Collision Avoidance (CSMA/CA), and token passing. Each of these access methods is uniquely structured to address the previously mentioned collision and data destination problems.

CSMA/CD

Carrier-Sense Multiple Access with Collision Detection can be categorized as a "listen," then "send" access method. CSMA/CD is one of the earliest developed access techniques and is the technique used in Ethernet, which is the network system Xerox Corporation developed whose technology has been licensed to many companies.

Under the CSMA/CD concept, when a station has data to send, it first listens to determine if any other station on the network is talking. If the channel is busy, the station waits until the channel becomes idle prior to transmitting data. Because it is possible for two stations to listen at the same time and discover an idle channel, it is also possible that the two stations could then transmit at the same time. When this situation arises, a collision occurs. On sensing that a collision has occurred, a delay scheme will be employed to prevent a repetition of the collision. Typically, each station uses either a randomly generated or predefined time-out period prior to attempting to retransmit the messages that previously collided. Because this access method requires hardware capable of detecting the occurrence of a collision, it is usually more expensive than the hardware required for other access methods.

The CSMA/CD access method is commonly used on baseband and broadband coaxial cable based networks. Although there are several versions of CSMA/CD marketed, by far the most common version is based on licensed technology from Xerox Corporation, with over 50 vendors developing products based on the vendor's Ethernet specifications. In an Ethernet network, data is bundled into packets ranging in size from 64 to 1518 bytes. Figure 15.6 illustrates the Ethernet packet format. Ethernet uses 8-bit bytes for transmission, with each packet containing an 8-byte preamble for synchronization consisting of a sequence of 64 bits of alternating ones and zeros. This is followed by

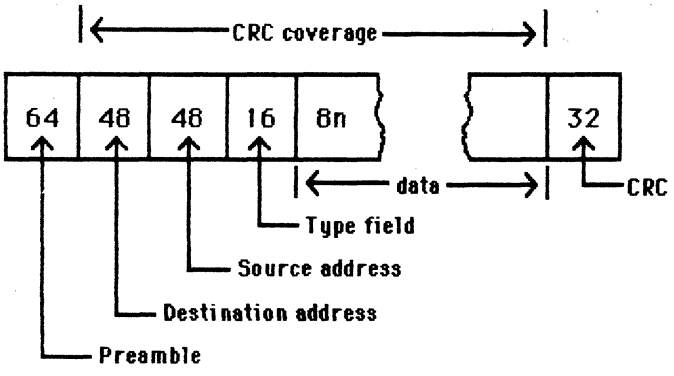


Figure 15.6
Ethernet Packet Format

two 6-byte addresses, the first representing the packet's destination, and the second identifying the originator of the packet. The 16-bit type field specifies how the data is to be interpreted and is similar to the control field employed in the SDLC protocol. By appropriate coding in the type field, the packet message will be denoted to contain supervisory data or information. The data field is byte-oriented and is represented by 8n in Figure 15.6, which means it is composed of one 8-bit byte times n bytes. The cyclic redundancy check (CRC) covers the destination through the data field and provides the error detection and correction mechanism to ensure data reaches its destination correctly.

The CSMA/CD access technique is best suited for networks with intermittent transmission, because an increase in traffic volume causes a corresponding increase in the probability of the cable being occupied when a station wishes to talk. In addition, as traffic volume builds under CSMA/CD throughput may decline, because there will be longer waits to gain access to the net as well as additional time-outs required to resolve collisions that occur.

CSMA/CA

Carrier-Sense Multiple Access with Collision Avoidance represents a modified version of the CSMA/CD access technique. Under the CSMA/CA access technique, each of the hardware devices attached to the talkers on the network estimates when a collision is likely to occur and avoids transmission during those times. Because this technique eliminates the requirement for collision detection hardware, the cost of hardware to implement this access technique is usually less than CSMA/CD hardware. In addition, with CSMA/CA vendors need not pay the licensing fees of Ethernet, which may be passed on to the user in the form of lower prices.

Token Passing

In a token passing access method, each time the network is turned on a token is generated. Consisting of a unique bit pattern, the token travels the length of the network, either around a ring or along the length of a bus. When a station on the network has data to transmit it must first seize a free token. The token is then transformed to indicate it is in use and is then appended to a packet of information being transmitted from one station to another. During the time the token is in use, the other stations on the network remain idle, eliminating the possibility of collisions occurring. Once the transmission is completed, the token is converted back into its original form and becomes available for use by the next station on the network. Because a station on the network can only transmit when it has a free token, token passing eliminates the requirement for collision detection hardware. This means that the cost of a token passing network is usually less than an equivalent CSMA/CD network. Due to the dependence of the network on the token, the loss of a station can bring the entire network down. To avoid this, several vendors have included special backup circuitry in their hardware, although this reduces the difference in price between token passing and CSMA/CD networks.

Due to the variety of transmission media, network structures, and access methods, there is no one best network for all users. Table 15.1 can help you compare the generalized advantages and disadvantages of the technical LAN characteristics, using the

Table 15.1
LAN Technical Characteristic Comparison

Characteristic	Transmission Medium		
	Twisted-Pair Wire	**Baseband Coaxial Cable**	**Broadband Coaxial Cable**
Topology	bus, star, or ring	bus or ring	bus or ring
Channels	single channel	single channel	multichannel
Data rate	up to 2 Mbps	2 to 10 Mbps	up to 400 Mbps
Maximum nodes on net	usually < 255	usually < 1024	several thousand
Geographical coverage	in thousands of feet	in miles	in tens of miles
Major advantages	low cost, may be able to use existing wiring	low cost, simple to install	supports voice, data, video applications simultaneously
Major disadvantages	limited bandwidth, requires conduits, low immunity to noise	low immunity to noise	high cost, difficult to install, requires RF modems and headend

Table 15.2
Technical Characteristics of Representative Networks

Vendor	Network	Transmission Medium	Data Rate (Mbps)	Access Method	Signal Type
AST Research	PC Net	twisted wire	0.8	CSMA/CD	Baseband
AT&T	StarLAN	twisted wire	1.0	CSMA/CD	Baseband
IBM	Cluster	coaxial cable	0.375	CSMA/CD	Baseband
IBM	PC Network	coaxial cable	2.0	CSMA/CD	Broadband
IBM	Token Ring	twisted wire, fiber optic	4.0	token	Baseband
Xerox	Ethernet	coaxial cable	10.0	CSMA/CD	Baseband
3COM	Ethershare	coaxial cable	10.0	CSMA/CD	Baseband

transmission medium as a frame of reference. Table 15.2 compares the technical characteristics of seven popularly used local area networks.

Hardware and Software Requirements

Besides cable and connectors, the installation of a local area network requires a variety of hardware and software products. As a minimum, each PS/2 on the network normally

requires the installation of an interface card into its system unit. This interface card contains a number of ROM modules and specialized circuitry as well as a microprocessor that implements the access method or protocol used to communicate on the common cable.

In some local area networks, one personal computer must be reserved to process the network commands and functions. Because it services these commands and functions, it is normally called the *network server.* A combination of specialized software that overlays the normal disk operating system (DOS) on each personal computer connected to the network as well as software placed on the server governs the operation of the network. To understand the role of the network server and the functions of LAN software, let us first review how a personal computer operates under a conventional version of DOS.

DOS is a single-user operating system designed to provide access to peripherals attached to the system as well as control of those peripherals, the interpretation of commands to perform various functions on the system, and management of disk storage and memory. Under DOS, your keyboard is the standard input device and your display is the standard output device, with the control of the personal computer limited to one user at a time.

As soon as a personal computer connected into a network is initialized, its network software routines are added to DOS, permitting the computer to interact with the rest of the network. Prior to the introduction of DOS Version 3.1, this software was normally an overlay to DOS that filtered commands. Thus, when a command was issued to perform a function on the PC, the software overlay permitted the command to pass directly to DOS for execution. If a command is issued that references the network, the software overlay intercepts or filters the command from reaching DOS and in conjunction with the adapter board transmits the command onto the network. If the network is server based, the nonlocal commands must be sent to a server for processing. The left-hand portion of Figure 15.7 illustrates the hardware and software components required when LAN software is designed as an overlay to DOS.

Prior to the introduction of DOS 3.1, most LAN vendors either developed proprietary methods to lock files and records or ignored incorporating such features, in effect limiting their networks to simple file swapping and printer sharing applications. Because there was no Network Basic Input/Output System (NETBIOS), a proprietary network BIOS

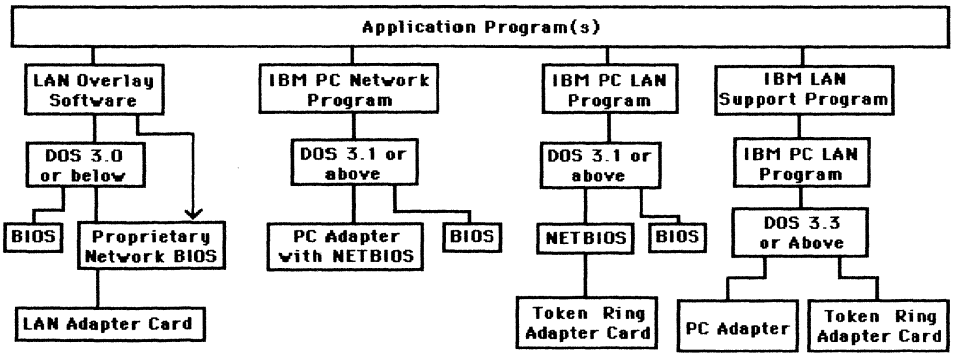


Figure 15.7 Potential PC LAN Hardware and Software Relationships

was developed and accessed via the vendor's LAN overlay software to send and receive data from the LAN adapter card. Here NETBIOS is the lowest level of software on a local area network, translating commands to send and receive data via the adapter card into the instructions in assembly language that actually perform the requested functions.

With the introduction of the IBM PC Network in August 1984, IBM released all three components required to implement an IBM local area network using IBM equipment—the IBM PC Network Program, PC DOS 3.1, and the IBM PC Network Adapter. The IBM PC Network Program was actually a tailored version of Microsoft Corporation's Microsoft Networks (MS-NET) software, which is essentially a program that overlays DOS and permits workstations on a network to share their disks and peripheral devices. DOS 3.1, also developed by Microsoft, added file and record locking capabilities to DOS, permitting multiple users to access and modify data without file corruption. Before file and record locking capabilities became part of DOS, custom software was required to obtain these functions, because otherwise the last person saving data in a file overwrote changes made by other persons to the file. Thus, DOS 3.1 provided networking and application programmers with a set of standards they could use in developing network software.

Included on the IBM PC Network Adapter card in ROM is an extensive set of programming instructions known as NETBIOS. The second column of Figure 15.7 illustrates the hardware and software components of an IBM PC LAN network when it was introduced.

When the IBM Token Ring Network was introduced, NETBIOS was removed from the adapter card and incorporated as a separate program activated from DOS. The third column of Figure 15.7 illustrates the initial hardware and software relationship for the IBM Token Ring local area network. Here, the network operating system for the Token Ring was renamed as the IBM PC LAN Program from its former name of the IBM PC Network Program.

Due to the standardization of file and record locking under DOS 3.1, any multiuser program written for DOS 3.1 will execute on any LAN that supports this version of DOS. Although DOS 3.1 supports many networking functions, it is not a networking operating system. In fact, a variety of network operating systems support DOS 3.1, including MS-NET, IBM's PC Network Program, IBM's Token Ring Program, and Novell's NetWare. This enables you to select a third-party network operating system to use with IBM network hardware. Alternatively, you can consider obtaining both third-party hardware and software to construct your local area network.

In 1987, IBM significantly revised the software required to operate its local area networks with the introduction of its LAN Support Program. This program replaced previously released software that provided network adapter support interfaces and the NETBIOS for the IBM Token Ring and the IBM PC Network. Although this program requires DOS 3.3 or a later version of the operating system, it adds interface support for both programs written to the NETBIOS interface as well as for programs written to the IEEE 802.2 interface, with the latter including IBM's Advanced Program-to-Program Communications (APPC). APPC support permits programs on separate PS/2s to communicate directly with one another, whereas non-APPC programs require communications between two workstations to be routed through an intermediate device, such as a host computer. Also included in the IBM LAN Support Program is a Con-

figuration Aid that installs device drivers in your computer's CONFIG.SYS file, based on your response to a series of panels that are displayed. The resulting device drivers cause specific files to be loaded based on your hardware and software requirements when your CONFIG.SYS file is executed. Table 15.3 lists the different device drivers the IBM LAN Support Program can place in your CONFIG.SYS file based on the type of network adapter used in your PC and the software interface support required. Listing 15.1 gives the resulting CONFIG.SYS file created by the IBM LAN Support Program based on the use of a Token Ring Network Adapter with IBM's APPC and NETBIOS software.

Listing 15.1 CONFIG.SYS File for a LAN

```
C>type config.bak
files=40
COUNTRY=001, 437
device=\LAN\DXMAOMOD.SYS 001
device=\LAN\DXMC1MOD.SYS
device=\LAN\DXMTOMOD.SYS 0=N S=9
SHELL=C:\COMMAND.COM /E:2000 /P
LASTDRIVE=Z
FCBS=16, 8

c>
```

Servers

Some networks require the use of a *dedicated server,* which means no local processing can be conducted on that device. In comparison, other networks operate with a non-dedicated server, which means that local processing can be conducted concurrently with server operations.

Servers can be further categorized as disk servers or file servers. A *disk server* is the simplest form of server and is basically a fixed disk that can be partitioned into volumes. Each volume appears to every user on the network as a disk drive would appear to a single-user system, that is, a disk drive letter designator. To the individual user, the remote disk drive operates in the same manner as if it were attached to his or her computer. When the shared disk is partitioned, volumes can be allocated as private or public. Typically, each personal computer on the network is allocated at least one private volume and access and storage of data in that volume is limited to the assigned PS/2. Then each network user would be insulated from the possibility of other users using or changing the data files located on his or her private volume. In comparison, public volumes can be shared by a number of personal computers and require more sophisticated software to prevent one user from accessing a file or a file record when it is already being accessed by another user. To simplify software, some local area networks only permit read only public volumes, whereas other networks permit read/write public volumes that incorporate several types of data access permissions.

Table 15.3
Device Driver Selection

Network Adapters to be Supported	Software Interface Supported: IEEE 802.2	IEEE 802.2 and NETBIOS	IEEE 802.2 and 3270 Wkstn Program	IEEE 802.2, NETBIOS, and 3270 Wkstn Program
Token-Ring Network: PC Adapter PC Adapter II Adapter/A	A0 C0	A0 C0 T0	A0 C1	A0 C1 T0
PC Network Adapter	A0 G2	A0 G2 T0	—	—
PC Network: Adapter II Adapter II/A Baseband Adapter Baseband Adapter/A	A0 G0	A0 G0 T0	A0 G1	A0 G1 T0

Legend:
A0 = DXMA0MOD.SYS
C0 = DXMC0MOD.SYS
C1 = DXMC1MOD.SYS
T0 = DXMT0MOD.SYS
G0 = DXMG0MOD.SYS
G1 = DXMG1MOD.SYS
G2 = DXMG2MOD.SYS

One of the major problems associated with local area networks is the limited amount of true multiuser application software available for use on such networks. Even when the network software has a file or record lockout capability, the application software may not have this capability. This means conflicts can easily arise in the shared use of a single-user program, forcing users to place individual copies of each program in their private volumes.

In comparison to a disk server network, in which the fixed disk is partitioned, a file server provides access to the entire disk to each user. Here access to the fixed disk is by individual file names. Although a file server requires more sophisticated software than a disk server, it enables two or more users to share access to any file on the disk.

IEEE 802 Standards

In 1985, the 802 Committee of the Institute of Electrical and Electronic Engineers completed a set of standards for local area networks. These standards govern the access method, transmission medium, and other LAN characteristics. Since the completion of the standards, they have been adopted by the American National Standards Institute, the National Bureau of Standards, and most LAN vendors, including IBM and AT&T.

The IEEE 802 local area network standards define several types of LANs based on different network access control methods and types of transmission mediums as illustrated in Figure 15.8. The Logical Link Control, defined as the IEEE 802.2 standard defines the exchange of data between devices attached to an LAN, including addressing formats.

At a lower layer, the IEEE standard defines three methods of access control. IEEE 802.3 describes a physical bus structure using CSMA/CD as the access method, which is essentially similar to the Xerox Ethernet LAN. The IEEE 802.4 standard defines a physical bus employing token passing as the access method. This standard has been adopted by General Motors as the data link layer of their Manufacturing Automation Protocol (MAP). IEEE 802.5 defines a physical ring network structure employing token passing as the access method equivalent to the IBM Token Ring Network.

Examining IBM LANs

Using the information previously presented in this chapter as a background in LAN operations and technology, next the discussion covers two IBM LAN products.

The original IBM Corporation product offering that permited PCs to be linked together was formally called a *Cluster*; however, this product was essentially a low-cost and limited local area network. The IBM Cluster was only marketed for approximately six months and was replaced by two IBM network products, the IBM PC Network and Token Ring Network. Each of these networks provides users with a much more sophisticated networking capability, although the increased sophistication comes at a higher cost. This section examines the IBM PC Network and Token Ring Network products to obtain an understanding of the use and constraints associated with each product.

IBM PC Network

The IBM PC Network was announced in August 1984, approximately six months after the IBM Cluster. It represents a more sophisticated local area networking capability

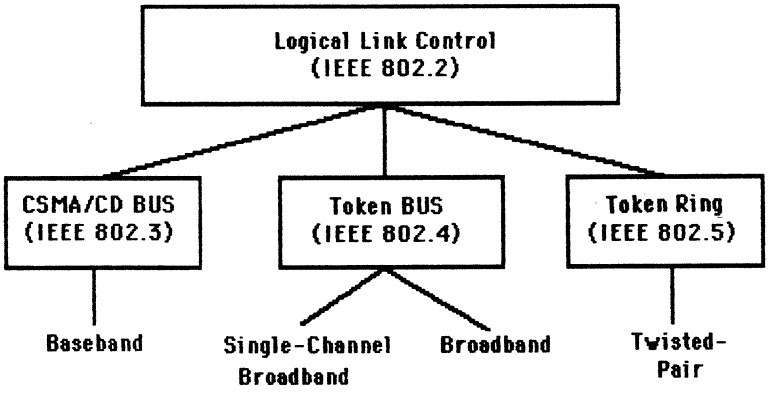


Figure 15.8
IEEE 802 LAN Standards

than the vendor's initial product. The PC Network is a 2 Mbps broadband coaxial cable system that employs the CSMA/CD access method.

Standard 75 ohm coaxial cable similar to standard CATV cable is used for the transmission medium of the network. Because this cable has a bandwidth of approximately 400 MHz of frequency and the actual data flows over a channel 6 MHz, it is possible to add voice, video, and other information onto the PC Network cable.

The PC Network is designed to link all members of the PC series and PS/2 family except the PCjr. Each PC connects to the common coaxial cable via a network adapter card that is manufactured for IBM by Sytek, Inc. The IBM PC Network Adapter II is designed for use in PC bus computers, including the original IBM PC series and the PS/2 Model 30. The IBM PC Network Adapter II/A is designed for use in Micro Channel based PS/2s, such as the Models 50, 60, 70, and 80. Each adapter card contains two Intel microprocessors and associated circuitry that provide the card's intelligence. Included on each adapter is an Intel 80188 microprocessor that operates the set of protocols known as LocalNet/PC. The LocalNet/PC protocols were originally developed by Sytek for that vendor's LocalNet 40 LAN, which was an expensive network designed for the minicomputer market. The second major chip on the adapter is an Intel 82586 chip that controls the CSMA/CD access method. Included on the adapter card is a modem that modulates and demodulates an analog carrier through frequency shift keying. Data is transmitted from a PC at 219 MHz and is received at 50.75 MHz. The former frequency is referred to as the *forward direction frequency,* while the latter is known as the *reverse direction frequency.*

One of the problems associated with broadband transmission on coaxial cable is the fact that high frequencies tend to attenuate more than lower frequencies. This means that incorrect cable lengths and uneven signal attenuation (called *cable tilt*) can cause a broadband network to fall out of tune.

In most broadband networks, an RF generator is used to transmit signals through a range of high to low frequencies to a receiver to determine whether a network is out of tune and, if so, by how much, so adjustments can be made to the network. IBM has eliminated the requirement to sweep and tune its network by employing fixed lengths of coaxial cable that are prebalanced with a built-in tilt compensation. Due to this design, connections are limited to certain combinations of 25-, 50-, 100-, and 200-foot coaxial cable segments that have built-in tilt compensation. An example of the constraints this can involve can be visualized by a requirement to connect two PCs 800 feet apart. The *IBM Technical Reference Manual* only permits four 200-foot segments to be used for this cabling requirement. Whereas this may appear insignificant, it does preclude the use of eight 100-foot segments or three 200-foot segments and two 100-foot segments, as well as other possible combinations, with the result that the planning and modification of a network requires careful consideration of the cable lengths.

In addition to the adapter card required for each PC on the network, a frequency translator or "head-end" is required. Known as the *Network Translator Unit,* this device receives all signals transmitted on the network at 219 MHz and translates them to 50.75 MHz. The PC Network requires one Network Translator Unit that can support up to 72 PC workstations. Although the PC Network is capable of supporting 1000 or more devices, under the terms of the marketing agreement between Sytek and IBM it appears that installations requiring more than 72 interfaces are referred to Sytek. In

addition, customized equipment may be necessary to support such nondata services, including voice and video.

Hardware Requirements

In addition to one PC Network adapter card required for each PC workstation, a minimum configured network requires one IBM PC Network Translator Unit (NTU) or head-end for the entire network and one or more cable kits to interconnect the various workstations to one another and the NTU. The NTU is provided with an 8-way splitter, which limits the number of workstations that can be interconnected to one another to eight in a minimum system as illustrated in Figure 15.9. Each personal computer, in turn, can be located up to 200 feet from the splitter. Thus, the maximum distance covered by a minimum system would be 400 feet.

To expand the PC Network to support a spanned distance in excess of 400 feet or to link more than 8 personal computers into the network requires the addition of one or more IBM Base Expander Kits and a combination of short-, medium-, and long-distance cabling kits.

Figure 15.10 illustrates how access to the Network Translator Unit is expanded to additional personal computers through the installation of one or more base expanders. First a T-connector is installed between the 8-way splitter and the NTU, permitting the first base expander to be connected to the network. The base expander in effect is another type of 8-way splitter to which one can connect a short (100-foot), medium (400-foot), or long (800-foot) distance kit. Because up to eight 8-way splitters can be connected to the base expander a maximum of 64 additional workstations can be added to the minimum network configuration, resulting in the capability of the PC Network to support a maximum of 72 workstations.

IBM PC Network Program

The IBM PC Network Program was introduced with the PC Network and is one of two network operating systems that can be used with the PC Network. The other network

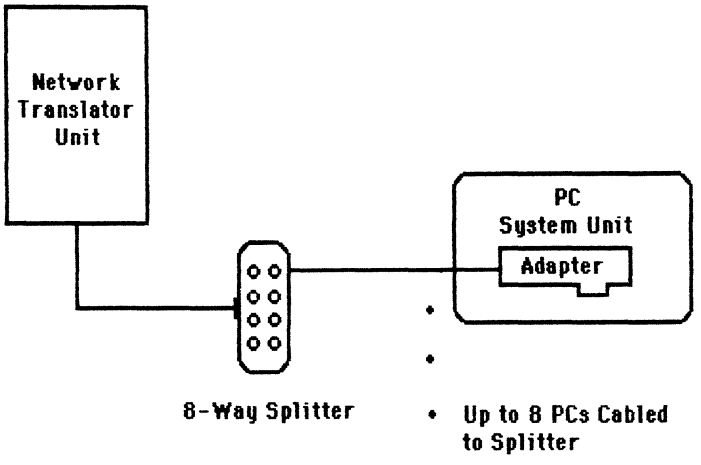


Figure 15.9
Minimum IBM PC Network

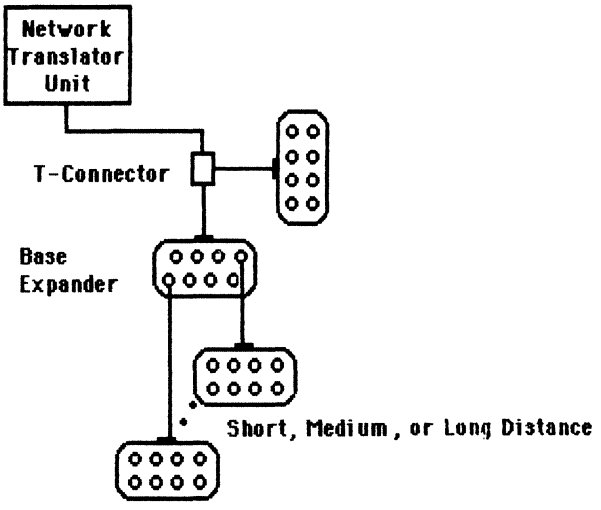


Figure 15.10
PC Network
Expansion

operating system is the IBM PC Local Area Network Program, which supports both the PC Network and the Token Ring Network. The IBM PC Network Program enables users to share the use of printers and disks and to communicate with other computers on the network. To facilitate its use, network users can give instructions to the PC Network Program through program menus or by issuing commands.

Computer and Device Names

Each computer attached to the network must be assigned a network name to identify it to the network. To distinguish network names from a directory and subdirectory they are prefixed with two backslashes (\\). Thus, \\GIL would reference the computer on the network assigned the name GIL, whereas \\GIL\DBASE would reference the directory DBASE on the computer named GIL.

Under the IBM PC Network Program, users can access shared devices once they know the network name of the computer with the shared device and the network name of the device they wish to use. Thus, if a letter-quality printer attached to the computer with the network name GIL was assigned the device name LTRPRINT, that printer could be accessed by any computer user attached to the network.

Hot Key

Similar to emulator products the PC Network Program has a hot key capability, which is actually a sequence of keys. By pressing the **Ctrl+Alt+Break** key combination, you toggle between the application you are executing and the PC Network Program.

DOS Usage

Users on the PC Network can execute most DOS commands. In addition, many DOS commands that reference other PCs in the network can be issued. For example, the command

```
COPY PAY.DBF \\GIL\DBASE
```

copies the file PAY.DBF from a default disk on one PC to the directory DBASE on another PC with the network name GIL.

PC Configurations

Each personal computer using the PC Network Program can be configured in one of four ways—as a Server, Messenger, Receiver, or Redirector. Table 15.4 compares the functional capabilities of each of the four PC configurations selectable with the PC Network Program. The key criteria in selecting the appropriate configuration are the hardware requirements for each configuration. As an example, a server requires one fixed disk, one diskette drive, and at least 320K bytes of memory. In comparison, a redirector requires one diskette drive and 128K bytes of memory.

Network Commands

Over 45 network commands are included in the PC Network Program. Similar to DOS commands, many network commands include optional parameters and switches. In addition, some network commands can be abbreviated. Table 15.5 lists 14 of the more common PC Network Program commands. As an example of the use of a parameter and switch in a network command, consider a network where it is desired to dedicate a PS/2 Model 50 as a server. Entering the command

```
NET START SRV GIL/TSI:00
```

causes no time slice increments to be allocated to local applications on the server named GIL, in effect dedicating that computer as a pure server. This means that other users accessing the server receive improved responses; however, the server is incapable of operating application programs concurrent with the server function.

Table 15.4
PC Network Program PC Configurations

	Configuration			
Function	**Server**	**Messenger**	**Receiver**	**Redirector**
Send Messages	Y	Y	Y	Y
Use Network Disks, Directories, and Printers	Y	Y	Y	Y
Receive Messages	Y	Y	Y	N
Save Messages	Y	Y	Y	N
Use Network Hot Key	Y	Y	N	N
Receive Messages for Other Computers	Y	Y	N	N
Transfer Messages to Other Computers	Y	Y	N	N
Share Disks, Directories, and Printers on Local PC	Y	N	N	N

Table 15.5
Common PC Network Commands

Command	Action
NET	starts PC LAN Program menus.
NET CONTINUE	restarts PC LAN on computer after a NET PAUSE.
NET ERROR	displays a list of network errors that have occurred on your computer.
NET FILE	displays current users and current record locks for a file as well as to close a file that is opened by other computers or that has locked records.
NET FORWARD	forwards or stops forwarding messages sent to your computer or another computer.
NET LOG	starts/stops placing messages you receive into a specified device or file or displays the status of logging.
NET NAME	allows additional names for which you want to receive messages to be displayed.
NET PAUSE	temporarily stops LAN functions on your computer.
NET PRINT	sends a file to a printer you are sharing with another network user or to a network printer you are using.
NET SEND	permits you to send messages to other users or all users.
NET SEPARATOR	controls whether a separator page is printed at the beginning of your print file.
NET SHARE	enables or disables other computers from using your devices or directories.
NET START	begins the PC LAN.
NET USE	specifies a device or a directory on a network computer that you want to use or stop using or to list the network devices you're using.

IBM Token Ring Network

Similar to the PC Network, every personal computer in a Token Ring Network must have an adapter card installed in an expansion slot in its System Unit. The original Token Ring Network Adapter card introduced with the Token Ring product announcement in October 1985 supports PC bus personal computers. This adapter card has been supplemented by two additional Token Ring Network Adapter cards. A second offering known as the Token Ring Network Adapter II contains double the 8K-byte buffer of the original adapter and is designed to provide increased performance for network servers since its additional memory enables the Adapter II to retrieve data from the network quicker than the original adapter. The third adapter is the Token Ring Network Adapter/A, which is designed for connecting Micro Channel PS/2s to an IBM Token Ring Network.

Topology

As previously mentioned, the Token Ring Network can actually be considered as a star-ring configuration, because personal computers are connected to a multistation access unit (MAU) and MAUs are then connected to one another to form larger networks. Up to eight PCs can be connected to one MAU as illustrated in the left portion of Figure 15.11.

Two Token Ring networks can be connected together by a bridge so both networks can logically function as one larger network. As illustrated in the middle of Figure 15.11, a personal computer, preferably a PC AT or a PS/2 Model 50Z, can be used as a bridge. When used as a bridge, the computer has two Token Ring Adapter II or Token Ring Adapter/A cards installed in its system unit. In addition, a special Token Ring Network Bridge Program must be obtained and the computer must be dedicated to operating this program.

A variety of Token Ring connectivity options are marketed by IBM that permit PCs on a Token Ring network to be connected to IBM mainframes and System/36 minicomputers. The right portion of Figure 15.11 illustrates how a Token Ring network can be connected to an IBM mainframe, such as a 43XX or 38XX computer system. In actuality, the MAU is cabled to the 3725 communications controller, with the communications controller requiring the installation of a Line Attachment Base (LAB) Type C and a Token Ring Interface Coupler (TIC). Because up to four TICs can be installed in one LAB, up to four Token Ring networks can be connected to one Type C LAB. Each personal computer on the Token Ring network requiring access to the mainframe must operate the IBM PC 3270 Emulation Program Version 3.0 or higher, which results in the PC emulating a 3274 control unit with an attached 3278/9 display and 3287 printer.

In the lower portion of Figure 15.11, a Token Ring network connection to an IBM mainframe through the use of an IBM 3174 controller is illustrated. Although six models of the 3174 controller have been announced, only three of these controllers offer Token

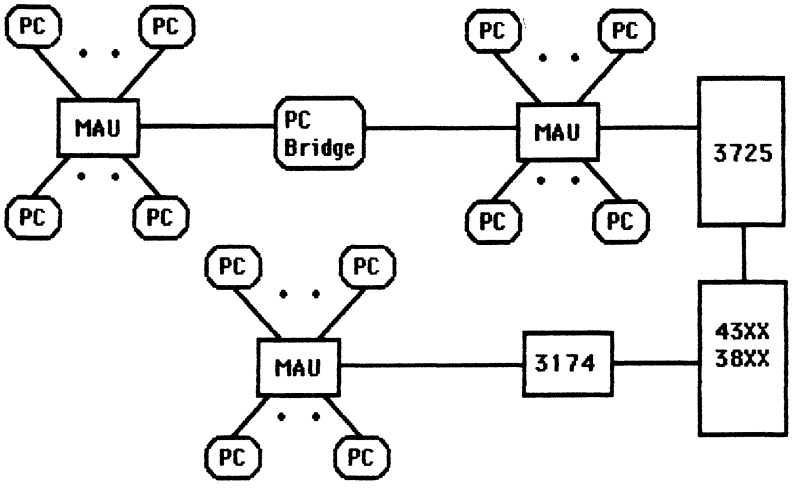


Figure 15.11
Token Ring Connection Options

Ring support. Two of the three controllers were designed as remote devices, with an RS-232 connection to the 3725 communications controller. The third controller is designed to be channel attached to an IBM mainframe as illustrated in the lower portion of Figure 15.11 and provides a gateway to the mainframe for a "local" Token Ring local area network.

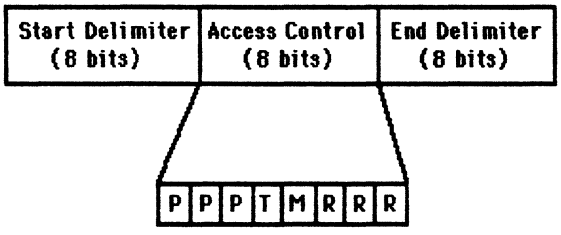
Token and Frame Formats

Two types of transmission formats are supported on the Token Ring network—token and frame. The token format as illustrated in the top of Figure 15.12 is the mechanism by which access to the ring is passed from one computer attached to the network to another device connected to the network.

Because the Token Ring network was designed to support different types of transmission, including synchronous 3270 full screen data transmission and asynchronous interactive transmission, a priority mechanism was incorporated to account for the

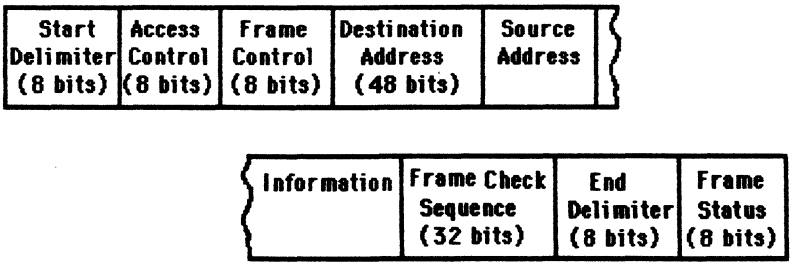
Figure 15.12
Token and Frame Formats

Token Format



Legend:
P = Priority bits
T = Token bit
M = Monitor bit
R = Reservation bits

Frame Format



differences in transmission requirements between different devices. Using three bits enables priorities to range from 0 to 7. When the token bit is set to 1 it indicates that a frame is on the network. Thus, a personal computer on the network must first seize a free token. Then, it can convert the token into a frame for transmitting data on the network.

The monitor bit is used to prevent a token with a priority exceeding 0 or a frame from continuously circulating on the Token Ring. This bit is transmitted as a 0 in all tokens and frames, except for a device on the network that functions as a monitor and thus obtains the capability to inspect and modify that bit. The reservation bits enable devices on the network to request the next token to be issued at the priority level of the device.

When a token is seized, the device with data to transmit develops a frame as illustrated in the lower portion of Figure 15.12. The first two bytes of the frame format in effect are the first two bytes of the token format, with the token bit now set to 1. The frame control field informs a receiving device on the network of the type of frame and how it should be buffered. Frames can be either *logical link control (LLC)* or *reference physical link* functions according to the IEEE 802.5 media access control (MAC) standard.

The destination and source addresses are each six bytes in length and identify the intended receiver and the sender of the frame. The frame check sequence field is four bytes in length and contains CRC error checking information that covers the frame control field through the information field.

The ending delimiter is a 1-byte field that ends both a token and a frame. In fact, when a token is seized its ending delimiter becomes the frame end delimiter. Finally, the frame status field is used by a receiving device to indicate that it successfully copied information from a frame transmitted to it. By the receiver modifying this byte, the sender receives information concerning the receiver's status.

Internal Versus External Resources

Under Extended Services resources, including application programs, data files and printers are categorized as internal or external. Internal resources are defined, controlled, and restricted for use within a domain. In comparison, external resources can be defined within a domain and used from a different domain or they can be provided from outside the domain of a user but used from within the user's domain.

IBM PC LAN Program

The IBM PC LAN Program supports both the PC Network and the Token Ring Network. This program is similar in functionality to the IBM PC Network Program, because you use it to share programs, data, and printer resources among PCs connected to either network. The IBM PC LAN Program also provides you with several enhanced features, including the ability to define and manage the resources of multiple servers as a single set of resources, from a single PC. The set of enhanced features included in the IBM PC LAN Program is called Extended Services, which was added to Version 1.3 of the program. With the introduction of that version of the PC LAN Program you can now select the use of Base Services or Extended Services.

Base Services

Base Services, as its name implies, consists of a set of basic functions you use to share resources. Base Services includes printer and file-sharing capability; communications among PS/2s; support for multiple servers; and messenger functions so you can send, receive, save, edit, and log messages to a file.

Base Services supports the four configuration types supported by the IBM PC Network Program—server, messenger, receiver, and redirector. Each personal computer using the IBM PC LAN Program Base Services has the same functionality capability as PCs using the PC Network Program. Refer to Table 15.4, which summarizes the functionality of the PC Network Program based on each of the four configuration types supported by that program. This functionality is the same as that incorporated into the IBM PC LAN Program Base Services.

Like the PC Network Program, IBM's PC LAN Program Base Services enables you to share resources by the use of menus or commands. You can use the menus by entering **NET** at the DOS prompt once Base Services are installed, or you can use network commands as previously covered in our discussion of the PC Network Program.

Extended Services

The key difference between Extended Services and Base Services is that the former is user-oriented, whereas the latter is machine-oriented. As such, Extended Services provides an environment of resources to be defined that one or more users employ for access to those resources without having to know the names of the resources. To obtain this added capability, Extended Services added several features and services that are not available in Base Services. Additions include domains, a system administrator, user identification (userids), passwords, internal and external resources, filesets, and Extended Services menus.

Under Extended Services, a *domain* is a grouping of one or more servers together with a definition of the resources provided by the servers and a set of users validated to log-on and use those resources. This structured environment is defined by a system administrator who is responsible for setting up and maintaining the domain. Functions performed by the system administrator include defining domain users, resources, and servers, as well as assigning user identifications, optional passwords, and fileset access.

Filesets

Under Extended Services a fileset is a directory on a server disk. The system administrator defines the name, contents, and location of each fileset, the type of user access (read only or update), and the users that can access the fileset. A special fileset called a *home fileset* is created by the system on a domain server. The home fileset contains information used to create your environment when you log on. This means you can have the same environment regardless of which workstation you use to access the network. Each user has read/write access to his or her home fileset, whereas all other users are barred access to that fileset.

Log-On Process

Figure 15.13 illustrates the log-on process to IBM PC LAN Program (PCLP) Extended Services. Once you have installed the IBM LAN Support Program and the IBM PC

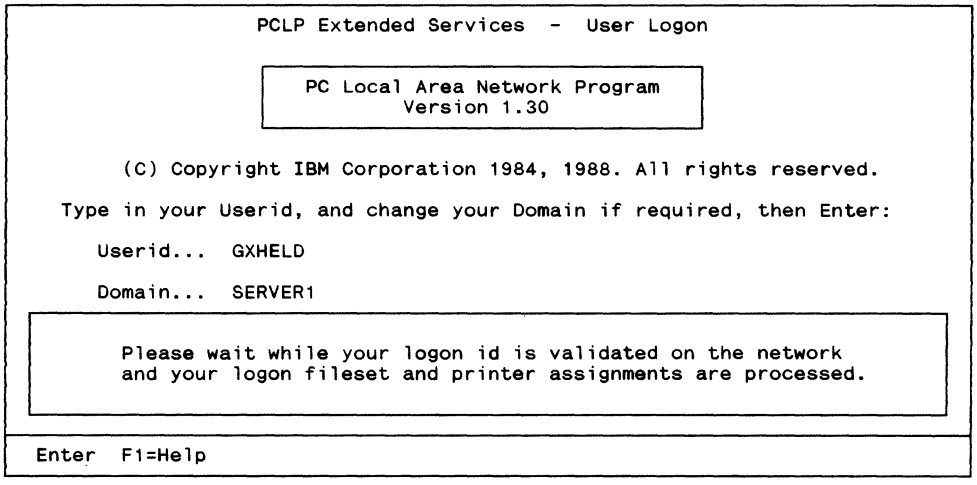


Figure 15.13
User Logon to Extended Services

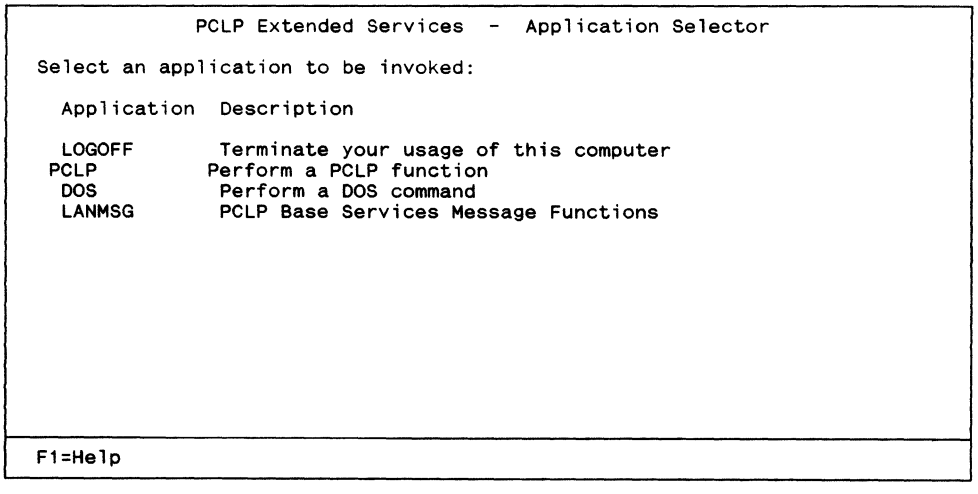


Figure 15.14
The Application Selector

LAN Program, each time you power-on your computer or perform a system reset you will be asked whether you want to use the PC LAN Program. If you respond affirmatively to the prompt, the PC LAN Program is loaded and, if your system was configured for Extended Services, the User Logon screen is displayed as illustrated in Figure 15.13. At this time you can enter your userid and domain to obtain access to the network.

Once you are successfully logged onto the network, the PCLP Extended Services Application Selector screen is displayed, as illustrated in Figure 15.14.

The Application Selector can be considered as the network's main menu. Entries on the Application Selector can be set up and modified by the system administrator or the user. To execute an application, you can use an arrow key to move a highlight bar to the appropriate application in the list and press **Enter**. After the system verifies that

you have access privileges to the desired application, it will make the required LAN connections and start the program.

Selecting LOGOFF logs you off the network and returns you to the log-on panel. The PCLP entry provides access to Extended Services, so you can print files, allocate or change filesets, change passwords, or access on-line reference information or an overview program. The latter provides basic information concerning the use of Extended Services. The resulting display when PCLP is selected from the Application Selector is shown in Figure 15.15.

Selecting the DOS entry from the Application Selector returns you to the DOS prompt; however, you remain logged onto the network. Figure 15.16 illustrates the screen display resulting from the selection of the DOS application listed in the Application Selector. Note that the current directory is Y, which is the home fileset directory of the server. By entering **C:**, you can change your current directory to your fixed disk. Otherwise, you can work with a directory on the file server.

The selection of the last entry on the Application Selector in Figure 15.14, LANMSG, takes you to the Base Services messaging system. A resulting set of displays lets you send, receive, and save messages as well as display messages previously saved on a file.

The initial display resulting from the selection of LANMSG is illustrated in Figure 15.17. Although the highlighted fields are not illustrated in Figure 15.17, there is one to the right of the Send the message to prompt. You can enter an asterisk into this field to broadcast the message to all network users, or you can enter a specific name

Figure 15.15
PCLP Menu

```
              PCLP Extended Services  -  PCLP Functions

Select a function:

  Function      Description

 PRINTF        Print a File
  USERSERV      Manage printer jobs and printers or list users
  FILEMAN       Allocate or Change Filesets
  USERADMN      Change passwords, LOGON allocations, applications used
  OLR           Online Reference Information
  OVERVIEW      Overview of PCLP Extended Services

 Esc=Quit  F1=Help
```

Figure 15.16
Accessing DOS From the Application Selector

```
Type EXIT to return to the Application Selector


The IBM Personal Computer DOS
Version 3.30 (C)Copyright International Business Machines Corp 1981, 1987
             (C)Copyright Microsoft Corp 1981, 1986

Y>
```

Figure 15.17
Initial Display Resulting From Selection of LANMSG

```
                                                          Base Services
                          Send Messages

Send the message to: (* for all computers)

Type message below :
```

```
Ctrl-Enter - Send message                Esc - Exit
Tab - Cursor to next field               Ctrl-PgDn - Erase message
F1 HELP/MORE KEYS    F3 ADJUST PARAGRAPH
F5 RETRIEVE FIRST    F6 RETRIEVE NEXT    F7 SAVE MESSAGE

                                                  Characters Free 1599
```

Table 15.6
Operational Capability and Transmission Media

Operational Capability	Unshielded Twisted Pair	IBM Cabling System Type 9 Cable
Capacity	72 devices	260 devices
Distance	about 300-600 feet	about 1000 feet
Speed	4 Mbps	4 Mbps
Attenuation	60 dB/km at 4 MHz (2 wiring closets maximum)	22 dB/1 km at 4 MHz (8 wiring closets maximum)

into the highlighted field. Next, you can enter a message up to 1599 characters in length and transmit the message to its destination by pressing **Ctrl+Enter**. As indicated in the lower portion of Figure 15.17, you can press one of the five function keys to perform specific functions, or you can press **Esc** to exit this menu. Performing the latter operation returns you to the Application Selector previously illustrated in Figure 15.14.

Media Variations

Both low-cost, unshielded twisted-pair and IBM Cabling System Type 9 cable can be used to interconnect personal computers on the Token Ring Network. Table 15.6 compares the operational capability of the Token Ring based on the two types of media that can be used. Because previously installed unshielded twisted pair may have many splices, the Token Ring's 4 Mbps signal can be significantly degraded when this medium is used unless you have prior knowledge about the condition of the wiring or use new

wiring. Due to IBM's emphasis on providing a variety of connectivity options for the Token Ring Network, this network can be considered to represent IBM's future networking strategies. In fact, although the PC Network may still be supported for awhile, it appears that those earlier products will shortly be replaced by the Token Ring Network.

16 / Reviewing Networking Strategies

The objective of this chapter is to review the primary methods whereby IBM PS/2s and compatible personal computers can be integrated into a corporate data communications network. Using the information presented in previous chapters of this book as a foundation, the personal computer networking strategies discussed as separate entities are appropriately grouped to indicate what your networking alternatives are.

Standalone Workstation

The most elementary networking strategy is to use the personal computer as a standalone workstation, providing each device with a communications program and modem. This enables you to use the personal computer for such functions as word processing and spreadsheet analysis while you have the capability to transmit and receive information between personal computers and host processors on an as-required basis. Depending on the type of network accessed and the communications software used, you may be able to communicate with a mainframe computer using full-screen features or access an information utility or mainframe on a line-by-line teletype basis. The typical standalone use of personal computers is illustrated in Figure 16.1.

Currently, the standalone personal computer configuration is the most common setup for accessing a communications network. In this configuration, access to the corporate network is primarily via asynchronous dial-up communications. If a file transfer capability is included in the emulator package, the company's host processor can be used as a store-and-forward mechanism for personal computer to personal computer file transfers. This permits the existing corporate communications network to be used to indirectly link personal computers together.

Although the standalone workstation is an economical method for linking personal computers, from a communications hardware and software perspective, your organi-

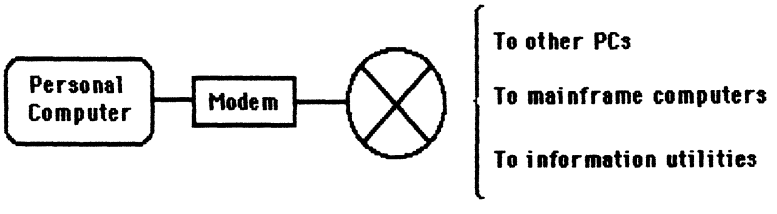


Figure 16.1
Using the PC as a Standalone Device

zation must have some type of "electronic mail" system on the host processor or develop a standard operating procedure to govern information transfers between PCs when the mainframe is used as a centralized repository. This ensures that the recipient can receive data from the originator. Because teletype emulation software does not emulate specific terminals, the personal computer may not be usable with certain host processor applications that are screen-oriented and use specific command codes to control the terminal's display.

Figure 16.2 illustrates the methods for obtaining line-by-line and full-screen access to a mainframe based on the type of communications software operating in an IBM PS/2. Assuming a TTY emulator is used, access to a mainframe is only obtainable on a line-by-line basis. If a full-screen terminal emulator communications program is obtained, such as a VT100 emulator, there are several methods whereby full screen access to the mainframe can be established. First, a packet network can be used to obtain both a data transportation facility and a protocol conversion facility. If the packet network does not perform protocol conversion, you can use either a standalone protocol

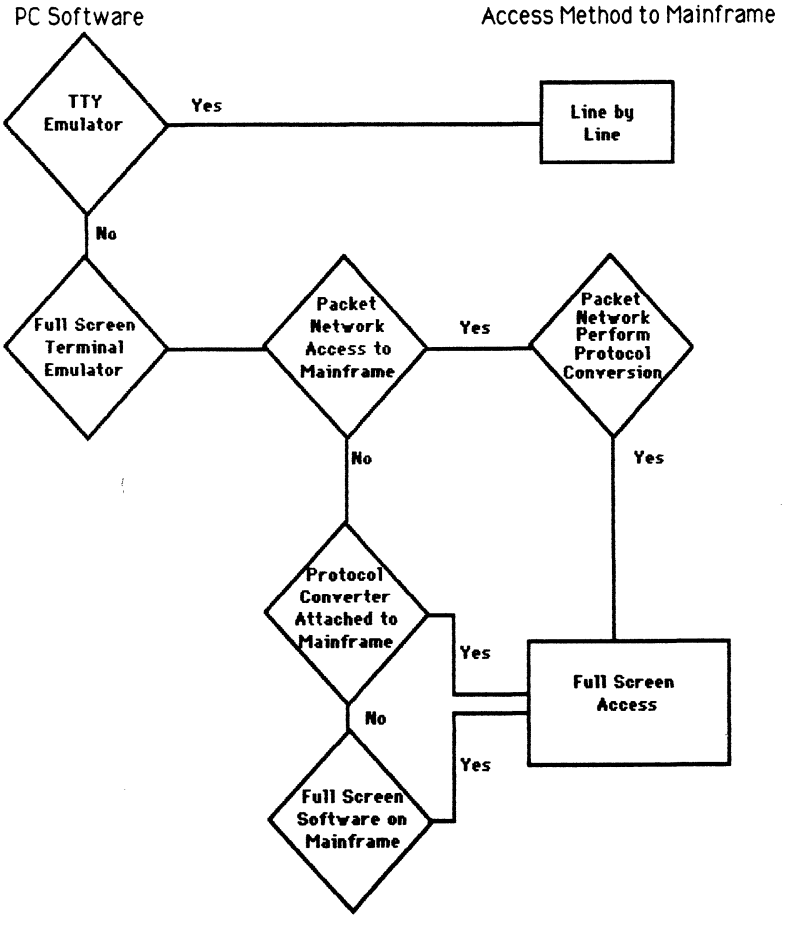


Figure 16.2 Line-by-Line and Full-Screen Access

converter attached to the organization's mainframe or a software program on the mainframe to operate in conjunction with the software operating on the personal computer.

Shared Workstation

Although a standalone configuration precludes sharing such peripherals as hard disks, modems, and printers when personal computers are located at a distinct geographical area, such peripherals can be shared if the computers are clustered with two or more devices at one location. When such situations arise, an elementary fallback switch can be employed so two personal computers can share a common printer, modem, or other peripheral, as illustrated in Figure 16.3.

As the figure shows, using a combination of fallback switches that cost approximately $100 per unit, two personal computers can share common peripherals, such as printers, plotters, or modems. This configuration permits each personal computer to communicate comparably to a standalone device; however, the cost of peripherals is distributed among two or more systems. The advantages and disadvantages of standalone personal computers that share peripherals are similar to individual standalone devices previously discussed. The only difference is in the sharing mechanism used. If each personal computer has an individual modem, a shared peripheral standalone unit functions similar to an individual standalone personal computer with respect to communications capability. When communications access is shared through a fallback switch, only one personal computer can communicate at any one time. Obviously, this reduces the number of network entry points to provide for such computers to access the corporate network.

3270 Access

A more sophisticated networking strategy is to use the personal computer as a 3270 type terminal replacement. Although the most common method to obtain 3270 access

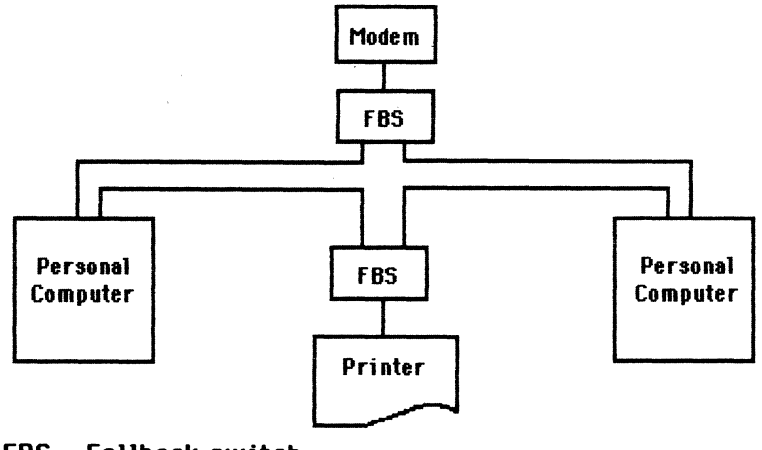


Figure 16.3
Sharing Peripherals in a Cluster

is by adding a synchronous communications adapter card into the system unit of a PS/2 and using a 3270 type terminal software emulator, you can also use a protocol converter, install specialized software on your mainframe computer, or employ the services of a value-added carrier to obtain this transmission capability.

Figure 16.4 illustrates the primary methods whereby an IBM PS/2 or compatible personal computer can gain access to a 3270 network on a full-screen basis. Because each of the access methods illustrated is dependent on the prior installation of hardware and software products, an organization's existing network structure normally has a large bearing on the access method or methods selected.

Due to the variety of hardware and software offered to enable 3270 access, PCs can obtain such access as an individual terminal or through a connection to a cluster controller. For both methods, the primary advantage of 3270 access is the inherent ability of the personal computer to interface with the applications on a full-screen basis, as well as the use of multiple keys on the personal computer to generate unique 3270 key codes for application program operations.

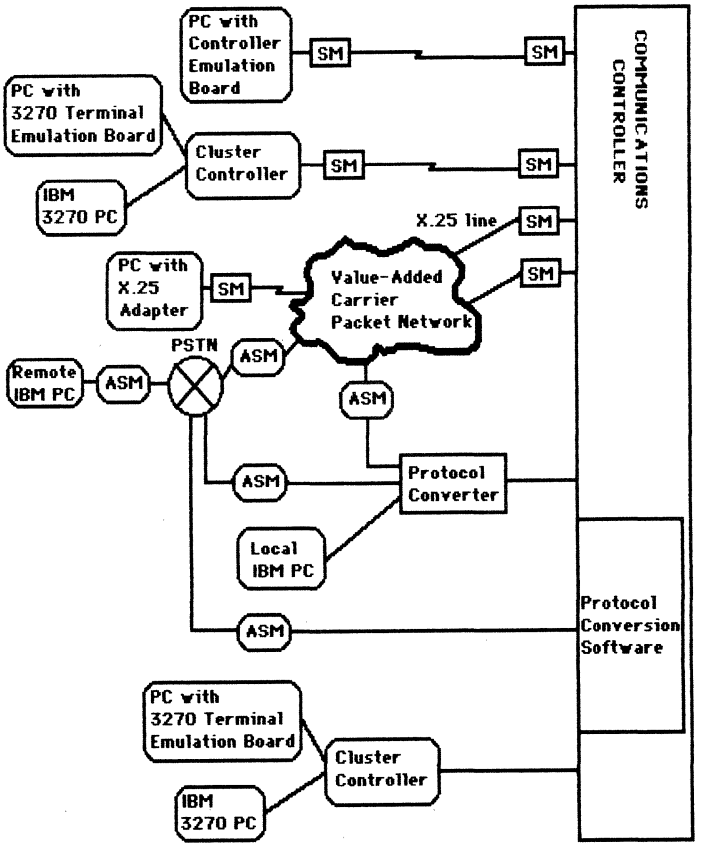


Figure 16.4
3270 Access Methods

Legend:
ASM = asynchronous modem
SM = synchronous modem

Another advantage of 3270 access includes the ability to replace 3270 type terminal devices with a personal computer. This provides local processing capability as well as the ability to transfer files. In comparison, a 3278 or 3279 terminal lacks both processing capability and file transfer capability.

If 3270 access is accomplished through a combination of hardware and software, such as the installation of an adapter board in the system unit of an IBM PS/2 or the use of a protocol converter, you must also add a modem, telephone line, and an interactive TTY emulator program if you require direct access to information utilities and other personal computers that transmit data asynchronously line-by-line. In comparison, if 3270 access is accomplished through a packet network that performs the data conversion, you can use the modem, telephone line, and emulator program to also access information utilities and other personal computers when 3270 access is not required, resulting in a degree of savings in hardware and software. This savings results from the fact that the packet network performs the required protocol conversion. This permits the personal computer to use a common serial port and asynchronous modem to access the organization's mainframe through the packet network as well as any required information utilities.

Normally, the only software requirement for 3270 access through a packet network is for a specific terminal emulator, such as a VT100 emulator program. Because some software emulator programs incorporate a simple line-by-line transmission option into the program, the program can also be used to access information utilities and other personal computers, as well as the packet network for 3270 access. Then you merely obtain one communications program and the popular asynchronous communications adapter to obtain both 3270 access and the ability to access information utilities and other PCs.

Local Area Network

Another networking strategy used by many companies is the integration of personal computers into a local area network (LAN). When equipped with an appropriate hardware adapter card and software, the personal computer becomes another network station on a local area network. This permits any personal computer on the LAN to communicate with any other network station, including peripheral devices, other personal computers, and host processors as illustrated in Figure 16.5. Currently, the cost of hardware and software to enable the linking of personal computers into LANs varies between $800 and $2000 per device.

The primary advantage of LANs is the capability to interconnect numerous devices via a common cable. This enables you to share peripherals among several network stations, as well as to disseminate common information throughout the network. Because the cost per connection exceeds the cost of most dot-matrix and many letter-quality printers, the economics of this networking strategy is, at best, marginally suitable if the objective is to share low-cost personal computer peripherals. Even if the local area network includes a gateway to enable the personal computers on the network to communicate with a host connected to the network, the personal computers may not be compatible with certain applications software. This is again because command codes

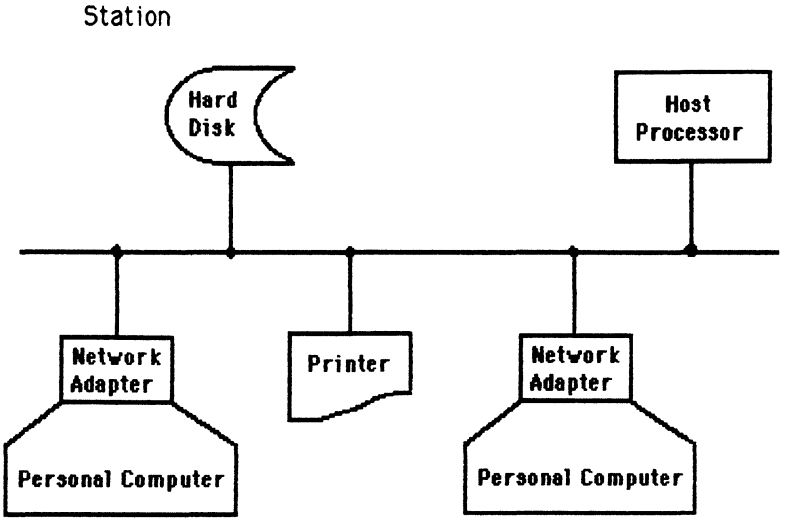


Figure 16.5 Using the Personal Computer as a Local Area Network Station

of the personal computer differ from those used by the application program. To obtain compatibility, your system may require a special workstation program whose cost should be considered in the overall network economic analysis. Remember to consider accessing a mainframe via a LAN versus using such other access methods as protocol converters, terminal emulators, or software.

Summary

The governing factor for developing the most appropriate networking strategy is knowing the communications application environment in which your personal computer is to be used. As indicated, there is no preferred networking strategy. There *is* an appropriate network strategy that can fulfill the communications application requirement of the personal computer in an economical and efficient manner.

To appreciate how an organization's existing network structure and the applications requirement will govern the selection of a networking strategy, consider the following examples. If you only desire to access an information network, the standalone strategy will suffice. If in addition to accessing an information network you want to replace an existing network terminal with a personal computer that is to be located near a control unit, the use of an asynchronous communications adapter in one expansion slot and a terminal emulator board in a second expansion slot can satisfy this dual requirement. Here the asynchronous communications adapter could be connected to a modem to obtain access to an information network, whereas the terminal emulator board could be connected to the control unit via coaxial cable to obtain access to the corporate mainframe.

As discussed, there are numerous hardware and software products you can use to enable personal computers to communicate on local area networks and with mainframes, information utilities, and nonnetworked personal computers. Prior to selecting a com-

munications strategy, you should define the application(s) that the personal computer is to perform. Next, you can perform a review of the organization's existing data communications network to determine both devices and software available for personal computer communications support. Using this information as well as knowledge of the networking options presented in this book helps you assess the variety of networking strategies you can use to satisfy the current networking requirements.

Index

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Here's a technical reference manual that provides an in-depth guide to the IBM PS/2 family of computers, DOS, OS/2, and data communications ...

IBM® PS/2®
USER'S REFERENCE MANUAL

This valuable technical reference is the most comprehensive, "nuts-and-bolts" guide available on the PS/2 family of computers, covering DOS 4.0 and 3.3, OS/2, and the OS/2 Presentation Manager. In the *IBM PS/2 User's Reference Manual,* you will also find thorough treatment of:

- Proper fixed disk organization for the efficient use of data storage space
- Creating and implementing configuration and batch files, including installation, hardware requirements, and commands
- OS/2 and the comparisons of DOS and OS/2 commands, as well as coverage of advanced DOS commands
- Data communications, including methods of transmission, Protocols, LANs, and networking techniques

Solutions to compatibility problems are provided, so you'll learn how to integrate PS/2s with other computers, and you'll find descriptions of third-party products that add functionality and increased performance to PS/2s. An abundance of illustrations demonstrate procedural steps.

If you're switching to PS/2 from the PC, and you're looking for a comprehensive reference guide to the IBM PS/2 system, the *IBM PS/2 User's Reference Manual* is a complete, clearly written reference that will help you quickly become proficient with the PS/2 system.

GILBERT HELD is the author of many successful microcomputer books, including *The IBM PC Upgrader's Manual, DOS Productivity Tips and Tricks,* and *Communicating with the IBM PC Series: Concepts, Hardware, Software, Networking.* He is the only person to have twice won the prestigious Karp Award for technical excellence in writing.

Cover Design: Jordana Roteman
Photo: FPG Int'l

JOHN WILEY & SONS
Business/Law/General Books Division
605 Third Avenue, New York, N.Y. 10158-0012
New York • Chichester • Brisbane • Toronto • Singapore

ISBN 0-471-62150

9 780471 621508