Exchange

Exchange

July 1985

Exchange of IBM PC Information

Exchange of IBM PC Information is a monthly publication of the National Distribution Division, International Business Machines Corporation, Boca Raton, Florida, USA.

Editor	Michael Engelberg
User Group Editor	Bernard Penney
Writer, Design Consultant	Karen Porterfield
Writer	John Warnock
Editorial Assistant	Steve Mahlum
Illustrators	John Alfred Dorn III Narda Lebo John Segal
Production	Cohen and Company
User Group Support Manager	Gene Barlow

Exchange of IBM PC Information is distributed at no charge to registered PC user groups. To register with us, please write to:

IBM PC User Group Support
IBM Corporation (2900)
P.O. Box 3022
Boca Raton, FL 33431-0922

To correspond with *Exchange*, please write to:

Editor, *Exchange*
IBM Corporation (2900)
P.O. Box 3022
Boca Raton, FL 33431-0922

POSTMASTER: send address changes to *Exchange of IBM PC Information*, IBM Corporation (2900), P.O. Box 3022, Boca Raton FL 33431-0922.

The IBM PC Enhanced Color Display

Paul Jones
Poughkeepsie IBM Club
Microcomputer Club
Poughkeepsie, New York

(Editor's note: The author's review of the IBM PC Enhanced Graphics Adapter was published in the April 1985 diskette Exchange.*)*

The IBM Enhanced Color Display (ECD) is the monitor designed by IBM for use with the Enhanced Graphics Adapter (EGA).

The ECD is easy to set up. It has a regular power plug, much like the original IBM Color Display, which cannot be plugged into the AC outlet at the back of the system unit. Its 9-pin D connector plugs into the back of the EGA card and is attached using a small screwdriver.

If your EGA arrives the way mine did, the switches and jumpers are not set up for the ECD, so you'll have to read through the EGA manual to set the switches properly. CAUTION: Don't forget to move the on-card jumpers to configure them for the ECD. The instructions are a bit misleading in that they tell you the switch setting for the EGA card, then for the system unit, and then back to the EGA for the jumpers. If you proceed step by step, everything should be fine. (By the way, if you are already using the Color/Graphics Monitor Adapter, you'll probably want to remove it unless your new EGA is going to drive an IBM Monochrome Display.)

Notice the small (about 1/4″) square hole in the back of the EGA card's retention plate. This hole allows access to the switch so that you can reconfigure the switch settings with the EGA still plugged into the system unit and the unit cover intact. This is useful because the ECD can run in two modes: normal color mode, which more closely resembles the standard color display for software compatibility purposes; and enhanced color mode, in which the display has 200 vertical lines for normal color and 350 lines for enhanced. I've set my EGA to enhanced color mode, and most programs work. If you have something that doesn't, try the normal color mode switch settings.

When the whole thing is set up, it looks quite good. On the original Color Display, the letters are obviously dots that are not very close together. However, on the ECD, the dots blend together much better. The ECD doesn't use quite as many dots as the standard Monochrome Display, but the bigger letter size more than compensates.

The boxes used to draw letters have these dimensions (in pixels):

Mono	9 wide by 14 high
Color	8 wide by 8 high
ECD	8 wide by 14 high

The mixing horizontal pixel in monochrome was mostly used for spacing, so the ECD letters may appear closer together. This "monochrome quality" text is automatic whenever you are in text mode (DOS, word processing, spread sheets, etc.). The EGA's higher resolution graphics requires newer software to take advantage of all the enhancements. Existing programs written for the Color/Graphics Monitor Adapter carry the old 8 x 8 text characters, even with the EGA.

Perhaps the most significant visual difference is due to the relative sizes of the screens. The following are corner-to-corner measurements of the displayable screens:

Mono	11.25 inches
Color	12 inches
ECD	12.5 inches

The ECD's extra inch or so appears to make the ECD slightly easier to read than the monochrome. I am quite pleased with the result; I had not anticipated it being this good.

The ECD is easy to read and the colors are good and clear.

An interesting improvement, due more to the EGA than the ECD, is the total lack of snow on screen writes. The old Color Display, especially if you used the POKE-type direct screen buffer writes, would jam the access with resulting little pixels of white, called snow. The alternative to snow was a very complex series of software instructions that wait for retrace intervals to write data, which resulted in slow screen updates. Another manifestation of this problem was the need for DOS and the Color/Graphics Monitor Adapter to turn off the display for scrolling. The result was not snow, but very bad blinking. With the new ECD and EGA, these inconveniences are things of the past. Not all programs take advantage of the speed potential—at least not yet—but the new screen never really snows or blinks.

I have both a Monochrome Display and an Enhanced Color Display connected to my PC. The DOS MODE command easily switches back and forth, but I have to admit I'll stay on the ECD. The default color mode for DOS text is white (or more accurately, gray) on a black background. Although this seems to be the classic color combination, I prefer green or yellow on black, and I'm sure other people have their own preferences. In any case, the color can be set via a simple BASICA program or through other means similar to those used to set colors on the IBM Color Display.

With the ECD and EGA in my system, I tried running a number of conventional color game programs to determine whether they were compatible. Thus far, I have had no compatibility problems. However, because the programs don't take advantage of the EGA's and ECD's capabilities, the display is not significantly better than the original Color Display.

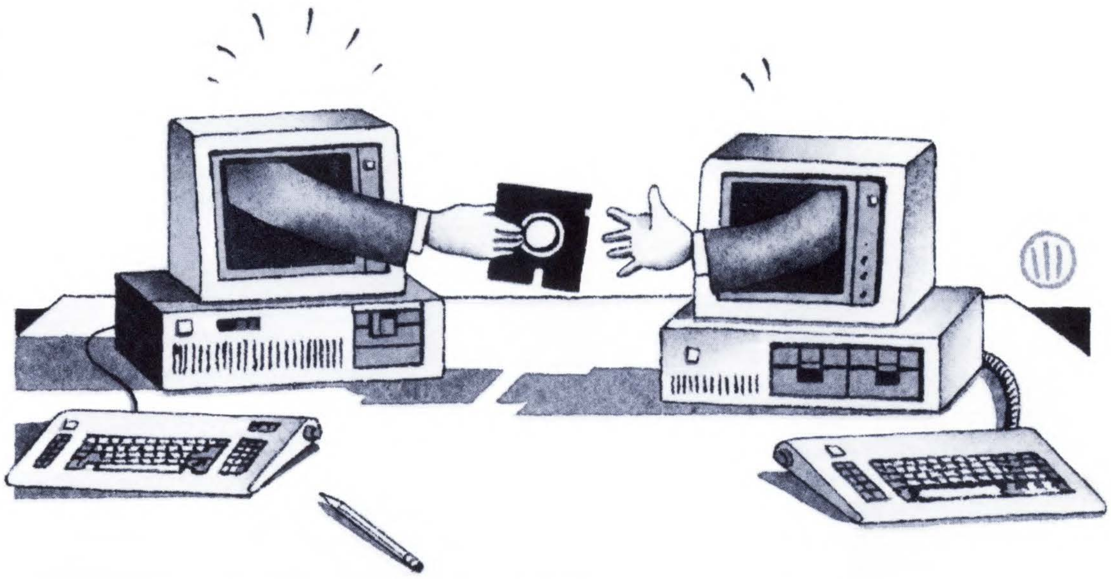
The Enhanced Color Display and Enhanced Graphics Adapter may not be for everyone. For the time being, there's a respectable amount of software that will neither run on the ECD or EGA nor take advantage of their new features. If you can live with these (hopefully) short-term problems, you'll get a technically superior display and adapter. The ECD is easy to read and the colors are good and clear. The resolution is good enough that you really need only this one monitor rather than two as in the past.

Programming the IBM Personal Computer AT

John Warnock
IBM Corporation

(Editor's note: This article is adapted from the IBM Personal Computer Seminar Proceedings.*)*

The IBM Personal Computer AT is available in two models. The base model contains:
—Advanced high-performance Intel 80286 Microprocessor
—Expansion socket for Intel 80287 Math Co-processor option
—64KB Read Only Memory containing:
  —Automatic power-on self test of system components
  —BASIC language interpreter
  —Basic Input/Output System (BIOS)
—256KB Random Access Memory
—Fixed Disk and Diskette Drive Adapter (Supports two 5¼″ Diskette Drives and two 5¼″ Fixed Disk Drives)
—1.2MB High Capacity Diskette Drive
—Real-Time Clock with CMOS RAM and battery backup
—Seven unused I/O expansion slots
—Speaker
—Keylock
—Enhanced keyboard
—Fixed-disk-in-use indicator (red)
—Power-on indicator (green)
—Worldwide 192 watt, 110/220 volt power supply
—Piggyback power cord
—Installation and Set-up manual and charts
—Guide to Operations manual with a diagnostic diskette and an "Exploring the IBM Personal Computer AT" diskette
—BASIC 3.00 manual



Address	Name	Function
000000 to 07FFFF	512KB System Board	System Board Memory
080000 to 09FFFF	128KB	I/O Channel Memory 128KB Memory Expansion Option
0A0000 to 0BFFFF	128KB Video RAM	Reserved for Video Display Buffer
0C0000 to 0DFFFF	128KB I/O Expansion ROM	Reserved for ROM on I/O Adapters
0E0000 to 0EFFFF	64KB Reserved on System Board	Duplicated Code Assignment at Address FE0000
0F0000 to 0FFFFF	64KB ROM on the System Board	Duplicated Code Assignment at Address FF0000
100000 to FDFFFF	Maximun Memory 15MB	I/O Channel Memory 512KB Memory Expansion Options
FE0000 to FEFFFF	64KB Reserved on System Board	Duplicated Code Assignment at Address 0E0000
FF0000 to FFFFFF	64KB ROM on the System Board	Duplicated Code Assignment at Address 0F0000

Figure 1: System Memory Map

The enhanced model includes everything in the base model plus the following:
—20MB Fixed Disk Drive
—Serial/Parallel Adapter (uses one I/O Expansion Slot)
—An added 256KB of memory (512KB total on system board)

Real-Time Clock Information
Figure 2 describes real-time clock bytes and specifies their addresses.

The Setup program initializes status registers A, B, C, and D when the time and date are set. Also, Interrupt 1A is the BIOS' interface to read/set the time and date. It initializes the status bytes the same as the Setup program.

Byte	Function	Address
0	Seconds	00
1	Second alarm	01
2	Minutes	02
3	Minute alarm	03
4	Hours	04
5	Hour alarm	05
6	Day of week	06
7	Date of month	07
8	Month	08
9	Year	09
10	Status Register A	0A
11	Status Register B	0B
12	Status Register C	0C
13	Status Register D	0D

Figure 2: Real-Time Clock Information (Addresses 00-0D)

IBM Personal Computer AT Compatibility

The IBM Personal Computer AT has a high level of programming compatibility with the IBM Personal Computers. It is possible to create Personal Computer AT software applications that can run without modification on other IBM Personal Computers.

To allow for maximum program compatibility, the IBM Personal Computer AT has maintained all BIOS system interrupts and utilizes the same IBM DOS interrupts. This means that applications which use the BIOS and the IBM DOS interrupts on the IBM Personal Computers operate the same on the IBM Personal Computer AT.

NOTE: The BIOS microcode of the IBM Personal Computer AT is not identical to that of other IBM Personal Computers. If an application bypasses the BIOS interrupt calls and directly accesses routines and/or storage locations in one system, it may not run in the other system. Some routines may be similar and some BIOS storage locations may be the same. It is strongly recommended that applications use only the BIOS and DOS interrupt interfaces in order to achieve compatibility among the IBM Personal Computer family.

Using the same language and the BIOS and DOS interfaces helps to achieve application compatibility. However, there still are several factors which need to be taken into consideration:

- **Timing Dependencies**
  Programs running on the IBM Personal Computer AT normally run faster than on the IBM Personal Computers. This is due to the higher performance processor (Intel 80286) and the higher bus transfer rate available on the IBM Personal Computer AT. The improved access time and transfer rate of the diskette and fixed disk drives also add to the overall performance improvement achieved on the IBM Personal Computer AT.

  It is strongly recommended not to depend on instruction execution speed to achieve specific application timing. The system timer can provide short interval timing for assembly language programs. Similar timing functions are available in BASIC.

  Performance of specific I/O devices may also differ between the IBM Personal Computer AT and the other IBM Personal Computers. You should also avoid using timing of any I/O device as a dependency for the application.
- **Unequal Configurations**
  In designing an application to run on both the IBM Personal Computer AT and other members of the IBM Personal Computer family, make sure that the required hardware configuration is available on all machines. This means the application's minimum requirements are met by all IBM Personal Computers.

Application Guidelines

The following information should be used to develop application programs for the IBM Personal Computer family.

High-Level Language Considerations

The IBM-supported languages of BASIC, FORTRAN, COBOL, Pascal and APL are the best choices for writing compatible programs.

If a program uses specific features of the hardware, that program may not be compatible with all IBM Personal Computers. The use of assembler language subroutines or hardware-specific commands (IN, OUT, PEEK, POKE,...) must follow the assembler language rules.

Any program that requires precise timing information should obtain it through a DOS or language interface; for example, TIME$ in BASIC. If greater precision is required, assembler techniques are available. The use of programming loops may prevent a program from being compatible with other IBM Personal Computers.

Assembly Language Programming Considerations

The following OP codes work differently on the IBM Personal Computer AT than they do on other IBM Personal Computers.

If the system microprocessor executes a POPF instruction, in either the real or the virtual address mode with CPL< =IOPL, then a pending maskable interrupt (the INTR pin active) may be improperly recognized after executing the POPF instruction even if maskable interrupts were disabled before the POPF instruction and the value popped had IF=0. If the interrupt is improperly recognized, the interrupt is still correctly executed. This has no effect when interrupts are enabled in either real or virtual address mode, and has no effect in the virtual address mode when CPL>IOPL.

The POPF instruction may be simulated with the code macro shown in Figure 3.

- PUSH SP pushes the current stack pointer. The microprocessor used in the IBM Personal Computer and the IBM Personal Computer XT pushes the new stack pointer.
- Single step interrupt (when TF=1) does not occur on the interrupt instruction (OP code hex CC,CD). The microprocessor in the IBM Personal Computer and the IBM Personal Computer XT does interrupt on the INT instruction.
- The divide error exception (interrupt 0) pushes the CS:IP of the instruction, causing the exception. The IBM Personal Computer and the IBM Personal Computer XT push the CS:IP following the instruction, causing the exception.
- Shift counts are masked to 5 bits. Shift counts greater than 31 are treated mod 32, that is, a shift count of 36 shifts the operand 4 places.

Assembly language programs should perform all I/O operations through ROM BIOS or DOS function calls.

- Program interrupts are used for access to these functions. This practice removes the absolute addressing from the program. Only the interrupt number is required.
- The Math Co-processor detects six different exception conditions that can occur during instruction execution. If the appropriate exception mask within the Co-processor is not set, the Co-processor sets its error signal. This error signal generates a hardware interrupt (interrupt 13) and causes the "BUSY" signal to the Co-processor to be held in the busy state. The "BUSY" signal may be cleared by an 8-bit I/O Write command to address hex F0 with D0 through D7 equal to 0.

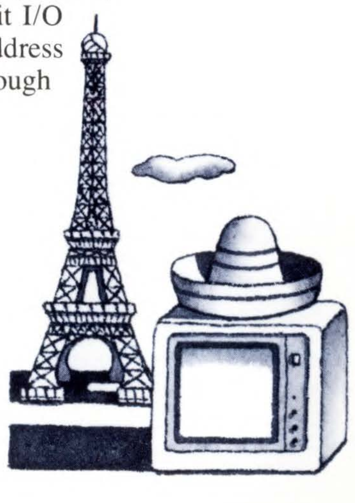
```
POPFF          MACRO          ;use POPFF instead of POPF
                              ;simulate popping flags
                              ;using IRET

EB 01          JMP $+3        ;jump around IRET

CF             IRET           ;POP CS, IP flags
0E             PUSH CS        ;push CS

E8 FB FF       CALL $-2       ;CALL within segment
                              ;program will continue here
```

Figure 3. POPF Code Simulation



The power-on-self test code in the system ROM enables hardware interrupt 13 and sets up its vector to point to a routine in ROM. The ROM routine clears the "BUSY" signal's latch and then transfers control to the address pointed to by the NMI vector. This allows code written for any IBM Personal Computer to work on an IBM Personal Computer AT. The NMI Interrupt handler should read the Co-processor's status to determine if the NMI was caused by the Co-processor. If the interrupt was not generated by the Co-processor, control should be passed to the original NMI Interrupt handler.

- Back-to-back I/O commands to the same I/O ports will not permit enough recovery time for I/O adapters. To insure enough time, a JMP SHORT $+2 must be inserted between IN/OUT instructions to the same I/O adapters.

  NOTE: MOV AL,AH type instruction does not allow enough recovery time. An example of the correct procedure follows:

  ```
  OUT IO_ADD,AL

  JMP SHORT $+2

  MOV AL,AH

  OUT IO_ADD,AL
  ```

- In the IBM Personal Computer AT, IRQ9 is redirected to INT hex 0A (hardware IRQ2). This insures that hardware designed to use IRQ2 will operate in the IBM Personal Computer AT.

Programs that use copy protection schemes should use the ROM BIOS diskette calls to read and verify the diskette and should not be timer dependent.

- There are several 80286 instructions that, when executed, lock out external bus signals. DMA requests will not be honored during the execution of these instructions. Multiple consecutive lock-out instructions will prevent DMA activity from the start of the first instruction to the end of the last consecutive lock-out instruction. In order to allow for necessary DMA cycles, as required by the diskette controller in a multitasking system, multiple lock-out instructions must be separated by a JMP SHORT $+2.
- The system can mask hardware sensitivity. New devices can change the ROM BIOS to accept the same programming interface on the new device.
- In cases where BIOS provides parameter tables, such as for video or diskette, a program may substitute new parameter values by building a new copy of the table and changing the vector to point to that table. However, the program should copy the current table, using the current vector, and then modify those locations in the table that need to be changed. In this way, the program will not inadvertently change any values that should be left the same.
- Disk_Base consists of 11 parameters required for diskette operation. They are pointed at by the data variable, Disk_Pointer, at absolute address 0:78. It is strongly recommended that the values supplied in ROM be used. If it becomes necessary to modify any of the parameters, build another parameter block and modify the address in Disk_Pointer to point to the new block. The parameters were established to operate both the High Capacity Diskette Drive and the Double Sided Diskette Drive. Three of the parameters in this table are under control of BIOS in the following situations. The Gap Length Parameter is no longer retrieved from the parameter block. Gap length used during diskette read, write, and verify operations is derived from within diskette BIOS. Gap length for format operations is still obtained from the parameter block. Special considerations are required for formatting operations. See the prologue of Diskette BIOS for the required details. If a parameter block contains a head settle time parameter value of 0 milliseconds, and a write operation is being performed, at least 15 milliseconds of head settle time will be enforced for a High

Capacity Diskette Drive and 20 milliseconds will be enforced for a Double Sided Diskette Drive. If a parameter block contains a motor start wait parameter of less than 1 second for a write or format operation or 625 milliseconds for a read or verify operation, Diskette BIOS will enforce those times listed above.

- The following procedure is used to determine the type of media inserted in the High Capacity Diskette Drive:
  1. Read Track 0, Head 0, Sector 1 to allow diskette BIOS to establish the media/drive combination. If this is successful, continue with the next step.
  2. Read Track 0, Sector 15. If an error occurs, a double sided diskette is in the drive. If a successful read occurs, a high capacity diskette is in the drive.
  3. If Step 1 fails, issue the reset function (AH=0) to diskette BIOS and retry. If a successful read cannot be done, the media needs to be formatted or is defective.

ROM BIOS and DOS do not provide for all functions. The following are the allowable I/O operations with which IBM will maintain compatibility in future systems.

- Control of the sound using port hex 61, and the sound channel of the timer/counter. A program can control timer/counter channels 0 and 2, ports hex 40, 42 and 43. A program must not change the value in port hex 41, because this port controls the dynamic-memory refresh. Channel 0 provides the time-of-day interrupt, and can also be used for timing short intervals. Channel 2 of the timer/counter is the output for speaker and cassette ports. This channel may also be used for timing short intervals, although it cannot interrupt at the end of the period.
- Control of the Game Control Adapter, port hex 201.

  NOTE: For delay on the paddle input, programs should use the timer rather than a program loop.

- Interrupt Mask Register (IMR), port hex 21, can be used selectively to mask and unmask the hardware features.

The following information pertains to absolute memory locations:

Use IBM-supported high level languages whenever possible.

- Interrupt Vectors (hex 0)—A program may change these to point at different processing routines. When an interrupt vector is modified, the original value should be retained. If the interrupt, either hardware or program, is not directed toward this device handler, the request should be passed to the next item in the list.
- Video Display Buffers (hex B0000 and B8000)—For each mode of operation defined in the video display BIOS, the memory map will remain the same. For example, the bit map for the 320 x 200 medium-resolution graphics mode of the Color/Graphics Monitor Adapter will be retained on any future adapter that supports that mode. If the bit map is modified, a different mode number will be used.
- ROM BIOS Data Area (40:0)—Any variables in this area will retain their current definition, whenever it is reasonable to do so. IBM may use these data areas for other purposes when the variable no longer has meaning in the system. In general, ROM BIOS data variables should be read or modified through BIOS calls whenever possible, and not with direct access to the variable.

A program that requires timing information should use either the time-of-day clock or the timing channels of the timer/counter. The input frequency to the timer will be maintained at 1.19 MHz, providing a constant time reference. Program loops should be avoided.

Programs that use copy protection schemes should use the ROM BIOS diskette calls to read and verify the diskette and should not be timer dependent. Any method can be used to create the diskette, although manufacturing capability should be considered. The verifying program can look at the diskette controller's status bytes in the ROM BIOS data area for additional information about embedded errors. More information about copy protection may be found under "Copy Protection" later in this section.

Any DOS program must be relocatable and insensitive to the size of DOS or its own load addresses. A program's memory requirement should be identified and contiguous with the load module. A program should not assume that all of memory is available to it.

Recommended Approaches
Use IBM-supported high level languages whenever possible. This usually provides object code compatibility (no action required) or source level compatibility (recompile and republish).

Always obtain timing for pacing the program or measuring elapsed time through DOS or language TIME functions.

Use only published DOS interfaces for all I/O and system services. Avoid the use of reserved or unpublished DOS functions and data.

Use only published BIOS interfaces for all system operations below the DOS level in Assembly Language programs. Avoid the use of reserved or unpublished BIOS functions and data.

Make certain that your program is relocatable.

Approaches To Avoid
Programs should not resort to techniques such as PEEK and POKE or make reference to any absolute memory locations.

Programs should not use instruction loops for time dependent operations.

Programs should not be machine sensitive unless the opportunity created by optimizing to a specific machine characteristic outweighs the opportunity lost as a result of limiting the program to specific IBM Personal Computer family members.

Documented Techniques
The IBM Personal Computer AT Technical Reference manual addresses several techniques for program portability. These include:

- A method to modify BIOS parameter tables by copying the provided table and changing those parameters which require change.
- Specific I/O operations which do not use BIOS: sound generation and game port operation.
- Video display buffers, ROM BIOS data area, interrupt mask register, interrupt vectors and some portability considerations.
- Copy protection approaches and incompatibility exposures.
- Multitasking considerations.
- An approach to the appropriate use of the new IBM Personal Computer AT SYS REQ key.

  NOTE: It is hoped that this approach will provide a consistent user interface and minimize conflict in "windowing" environments.

- A method to determine the media type in the High Capacity Diskette Drive.

IBM Personal Computer AT Differences
The following differences between the IBM Personal Computer AT and the IBM Personal Computer XT are discussed in the IBM Personal Computer AT Technical Reference manual.

There are some differences between how instructions operate in the 8088 (IBM Personal Computer, IBM Personal Computer XT, IBM Portable Personal Computer) and the 80286 (IBM Personal Computer AT). The following summarizes these differences:

- The POPF instruction may recognize pending interrupts even if interrupts are masked.
- The PUSH SP pushes the current stack pointer rather than the new stack pointer as in the 8088.
- The single step interrupt does not occur for certain instructions.
- A divide error exception pushes the pointer of the instruction causing the exception rather than the pointer of the next instruction, as was done in the 8088.
- Shift counts of greater than 31 are treated modulo 32 (e.g., shift 36 shifts 4 places).
- The MOV instruction in the format MOV CS,XX (where XX is a 16-bit register or memory location) is not supported on the Intel 8088, but it works. This instruction does not work in the Intel 80286. This is covered in the new Technical Reference manual.
- The 80286 has additional instructions which are not available in the 8088.
- There are differences between the Intel 80287 Math Co-processor and the Intel 8087 Math Co-processor found in the other members of the IBM Personal Computer family.

Any program which executes "back to back" I/O instructions to the same port may not execute the second I/O due to the speed of the new processor versus the speed of the I/O adapters.

Results Of Compatibility Testing
IBM began compatibility testing early in the development cycle of the hardware and software. As a result, many incompatibilities were removed.

Over 75 percent of the applications tested work on the IBM Personal Computer AT using DOS 3.00. The programs which do not work properly fall into four main categories:

1. Most of the programs which use instruction loops for timing do not work satisfactorily on the IBM Personal Computer AT.
2. Some programs use non-published interfaces which IBM could not maintain in the new system.
3. Some programs dwell on the physical characteristics of the system with the objective of being extremely user friendly. To the extent that specific keyboard locations and the diskette drive location changed, some of these programs are confusing and may no longer satisfy the objective of ease-of-use.
4. Some programs did not work because of differences between the 80286 and the 8088.

The IBM Personal Computer Network

(Part 2)

John Warnock
IBM Corporation

(Editor's note: This is the second part of an article about the IBM Personal Computer Network. This part describes the IBM Personal Computer Network Program. Part 1, published last month, discussed the hardware and a description of the network architecture. This article is adapted from the IBM Personal Computer Seminar Proceedings.*)*

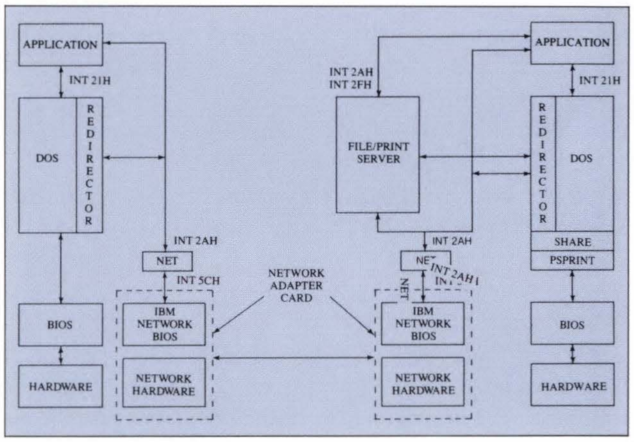


Figure 1. Network Overview

The IBM PC Network Program enables multiple IBM Personal Computers to be used in a local area network configuration. The PC Network Program supports the interconnection of IBM Personal Computers, IBM Portable Personal Computers, IBM Personal Computer XTs and IBM Personal Computer ATs, each with an IBM Network Adapter.

The IBM PC Network Program provides a range of configurations for users who need different levels of function. It also provides three levels of interface to the network: an easy-to-learn, full-screen, menu-oriented interface; a DOS command line interface; and a program interface for application developers.

Figure 1 gives a high-level overview of the components of the IBM PC Network Program and DOS 3.10, which supports the PC Network.

Network Program Configurations
The IBM PC Network Program resides in each personal computer on the network and can be configured in four different ways. Depending on user requirements, a personal computer can perform any one of the four networking roles:

- **Redirector**—reroutes local file and print I/O requests (interrupts 21H and 17H) over the network to a Server computer (described below). This makes the disk files and printers of the remote Server computer appear to be attached directly to a local personal computer. Network commands can be issued from the DOS command line. The full-screen redirector interface runs as an application on the local personal computer.

- **Receiver**—in addition to the Redirector function, a Receiver receives network messages and routes them to the console, printer or disk file.
- **Messenger**—In addition to Redirector and Receiver functions, a Messenger has a full-screen editor for messages. It adds names for the reception of messages and forwards those names to the other computers. With a Messenger function, you can switch back and forth between an application program and the Messenger's full-screen interface.
- **Server**—In addition to the functions of the three configurations above, a Server shares its disks, diskettes, directories and print devices with other personal computers. While it is serving other computers in the network, a Server computer can also run applications.

In summary, the PC Network Program lets users simultaneously share the use of disks, directories and print devices on the network; send, receive and log messages; and do simple file transfers. In this manner, one personal computer in the network can handle the files or do printing for another personal computer in the network. You do not need a dedicated personal computer for centralizing files or printing.

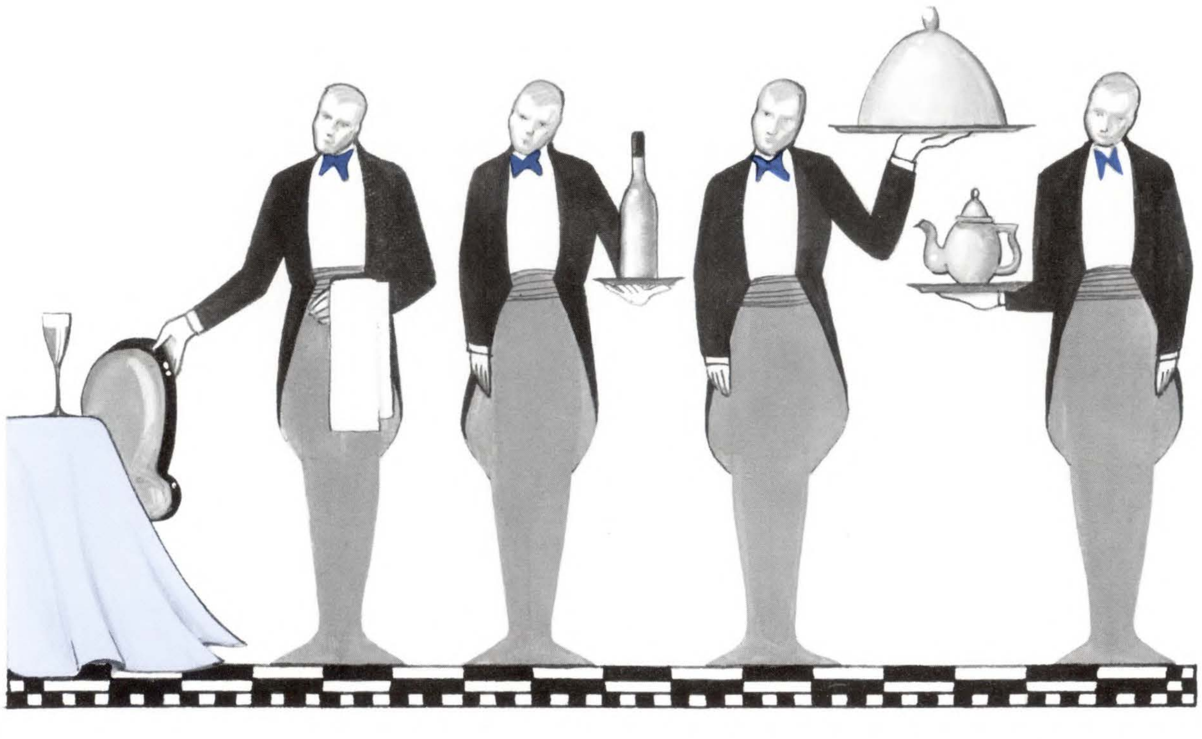
Servers
Using the PC Network Program, any fixed disk system—IBM Personal Computer AT, IBM Personal Computer XT or IBM Personal Computer with an Expansion Unit—can be designated as a Server.

When a remote user accesses a file/print server's disk, DOS 3.10 makes it appear as though that disk is attached to the remote user's personal computer. A Server disk can have access rights (read, write, create), access restrictions and a password. This makes it possible to run application programs that were not originally designed for network use.

A Server can make use of up to three printers on the network. Each printer can be password protected. The Server adds print jobs to a queue and directs them to an available printer. The print queue can hold 100 files. The Server's operator can examine and modify the print queues and can control the operation of the printers. Remote users can examine the status of their print files on the Server.

A single-session version of the Server permits file transfer but not concurrent execution of applications. A remote user who is willing to share files can give access to other users. Accessing users establish communications as they would when communicating with a normal Server. Multiple Servers can be active in one network at the same time.

Network Interface
The IBM PC Network provides three ways to access the network functions. The simplest way is an easy-to-learn, full-screen, menu-oriented operator interface that uses function keys, help screens and a nontechnical vocabulary. The second way is a DOS-like command line interface for faster operation and BATch file processing. Finally, there is a program interface with low-level sharing control and network status for application developers. Figure 2 shows how these interfaces are related.



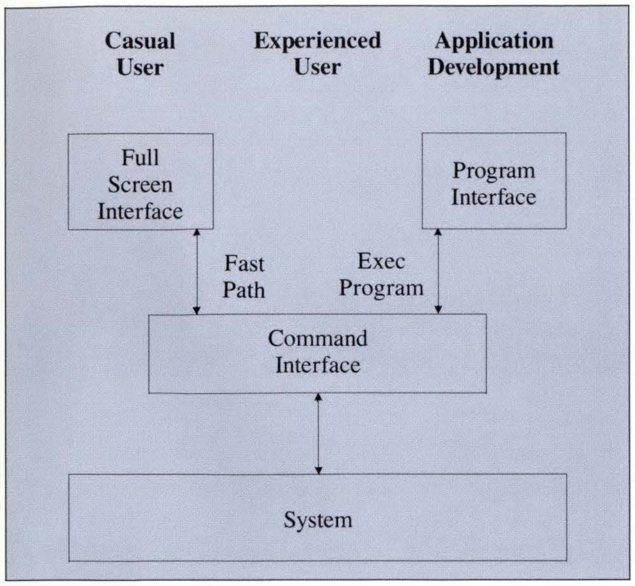
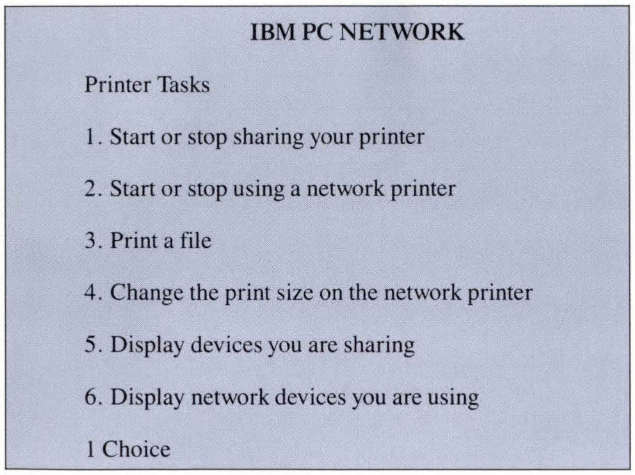


Figure 2. Network Interfaces

Full Screen Interface

The full screen interface is a stand-alone application with menu screens such as the one shown in Figure 3. It also provides prompted fill-in-the-blank entries when needed.



IBM PC NETWORK

Printer Tasks

1. Start or stop sharing your printer
2. Start or stop using a network printer
3. Print a file
4. Change the print size on the network printer
5. Display devices you are sharing
6. Display network devices you are using

1 Choice

Figure 3. Full Screen Network Interface

Command Interface

The command interface lets you control the network using the DOS command line. It is faster and more direct than going through the menus of the full screen interface. The following commands and functions are provided:

Command	Function
NET START	Start up one of the four Network Program configurations
NET SHARE/NET USE	Share and use network resources
NET PAUSE/ NET CONTINUE	Suspend and resume network operations
NET SEND	Send message(s)
NET LOG	Log message(s) to display, printer or file
NET NAME	Add a name for receiving messages
NET FORWARD	Forward a name to another computer so that messages can be received there
NET PRINT	Print a file on a network printer or get status of print queue
NET SEPARATOR	Define a separator page to be printed between print files
NET ERROR	List the network error log
NET FILE	List the current users and current record locks for a file
MODE	Set the character and line spacing for a network printer
PERMIT	Stand-alone single-session version of the Server

Program Interface

The program interface to the IBM PC Network Program is a set of interrupt calls that provides information about installation, redirection control and application programs which use the network facilities. The following chart lists the functions by interrupt and register codes.

	Register		
Interrupt	AH	AL	Function
2FH	BB	00	Net command installation check
	BB	03	Get server post address
	BB	04	Set server post address
21H	3D		Open file with sharing specified
	44	09	IOCTL: Is device redirected?
	44	0A	IOCTL: Is handle local or remote?
	44	0B	IOCTL: Change sharing retry count
	59		Get extended error
	5A		Create temporary file with unique name
	5B		Create new file
	5C	00	Lock byte range
	5C	01	Unlock byte range
	5E	00	Get name of computer
	5F	02	Get assign list entry
	5F	03	Redirect device to net
	5F	04	Cancel redirection
2AH	00		Installation check
	01		Execute NETBIOS request
	02		Set net printer mode
	03		Get device shared status

Server/Redirector Protocol
Servers and remote computers communicate control information and data across the network using the Server/Redirector Protocol. The primary control blocks in the protocol are the Server Message Block (SMB) and SMB Data Block:

```
SMB             STRUC
                DB              0FFh, 'SMB'  ;SMB Block Identifier
                                0            ;Tag
SMB FUNCTION    DB              0            ;Function code
SMB RETCLASS    DB                           ;Return error class
                                             ;Values 00-7F and 0FFH
                                0            ;Reserved
SMB RETCODE     DW              0            ;Return error code
SMB HEINFO      DB                           ;AH value on INT 24H
                                0            ;Else reserved
SMB RESV1       DB              0            ;Reserved, must be zero
SMB RESV2       DW              0            ;Reserved, must be zero
SMB RESV3       DW              0            ;Reserved, must be zero
SMB RESV4       DW              0            ;Reserved, must be zero
SMB RESV5       DW              0            ;Reserved, must be zero
SMB RESV6       DW              0            ;Reserved, must be zero
SMB RESV7       DW              0            ;Reserved, must be zero
SMB RESV8       DW              0            ;Reserved, must be zero
SMB NPID        DW              0            ;Network path ID
SMB PID         DW              0            ;Process ID
SMB RESV9       DW              0            ;Reserved, must be zero
SMB RESV10      DW              0            ;Reserved, must be zero
SMB PARMCNT     DB              0            ;Count of parms
SMB P1          DW                           ;Function-dependent
                                .            ;word
.    .          .               .            ;parameters
.    .          .               .            ;
.    .          .               0            ;
SMB Pn          DW              0            ;(0 or more)
SMB BUFLEN      DW                           ;Length of buffer for data
SMB BUF         EQU THIS BYTE                ;Buffer area
                SMB ENDS
SMB DATA        STRUC                        :ID of item
SMB ID          DB 0                         :(Values 00H TO 7FH
                                             ;reserved)
SMBSDB          EQU 1                        ;Item is a server data
                                             ;block
SMBDIALECT      EQU 2                        ;Item is a dialect ID string
                                             ;(Protocol identifier)
SMBPATH         EQU 3                        ;Item is an ASCIIZ string
                                             ;(Reserved for future use)
SMBASCIZ        EQU 4                        ;Item is an ASCIIZ string
SMBVARLEN       EQU 5                        ;Item is function-specific
                                             ;format: like SDB
SMB DATUM       EQU THIS BYTE                ;Buffer area
                SMB DATA ENDS
```

Note: All word values are stored in (low, high) order.

For types such as SMBDIALECT, SMBPATH, SMBBASCIZ, the format is:

```
DB          type            ;Type ID
DB          'string'        ;Actual data
DB          0               ;End marker
```

For type SMBSDB (Server Data Block), the format is:

```
SDB             STRUC
SDB ID          DB              SMBSDB      ;ID byte
SDB BUFLEN      DW              0           ;Length of
                                            buffer area
SDB BUF         EQU THIS                    ;Buffer area
                BYTE
SDB             ENDS
```

File Sharing

DOS 3.10 provides an extensive range of modes for sharing network applications, including a Compatibility mode for applications written without networking in mind.

Compatibility Mode
A file is in Compatibility mode if the file is opened by:

- Any of the CREATE function calls:

16H	Create File (FCB)
3CH	Create a File (handle)
5AH	Create Temporary File
5BH	Create New File

- An FCB function call:

0FH	Open File

File control block (FCB) function calls 0DH through 24H support DOS 1.10 files. The file control block makes the application maintain the information normally maintained in the directory entry.

- A handle function call with Compatibility mode specified:

3DH	Open a File

Extended file control block (handle) function calls 39H through 57H are recommended for controlling files. They permit access to files outside the current directory, do not require the application to maintain the directory entry, and allow access to the attribute byte.

A file can be opened any number of times in Compatibility mode, as long as the file is not currently open under one of the four sharing modes. If the file is marked read-only, and its sharing mode is Deny Write with Read Access, the file may be opened in Compatibility mode with read access. If the file was successfully opened in one of the other sharing modes and an attempt is made to open the file again in Compatibility mode, an interrupt 24H error will indicate "Drive not ready," and the extended error will indicate that there was a "Sharing violation."

DOS changes the sharing modes for a file opened in Compatibility mode, depending on the read-only attribute of the file. This permits sharing of read-only files.

Read-only file attributes are:

File Opened By	Access	Sharing Mode
FCB	Read/Write	Deny Write
Handle: Read	Read Only	Deny Write
Handle: Write	Error	—
Handle: Read/Write	Error	—

File attributes other than read-only are:

File Opened By	Access	Sharing Mode
FCB	Read/Write	Compatibility
Handle: Read	Read/Write	Compatibility
Handle: Write	Write	Compatibility
Handle: Read/Write	Write	Compatibility

Deny Read/Write Mode (Exclusive)
If a file is opened successfully in Deny Read/Write mode, access to the file is exclusive. A file currently open in this mode cannot be opened again in any sharing mode by any process (including the current process) until the file is closed.

Deny Write Mode
A file successfully opened in Deny Write mode prevents any write access opens to the file until the file is closed. An attempt to open a file in Deny Write mode is unsuccessful if the file is currently open with a write access.

Deny Read Mode
A file successfully opened in Deny Read mode prevents any read access opens to the file until the file is closed. An attempt to open a file in Deny Read mode is unsuccessful if the file is currently open in Compatibility mode or with read access.

Deny None Mode
A file successfully opened in Deny None mode places no restrictions on the read/write accessibility of the file. An attempt to open a file in Deny None mode is unsuccessful if the file is currently open in Compatibility mode.

The following file sharing matrix shows the results of opening, and subsequently attempting to reopen, the same file using all combinations of access and sharing modes.

			Second, Third, ... Open											
			Deny Read/ Write (DRW)			Deny Write (DW)			Deny Read (DR)			Deny None (All)		
			I	IO	O	I	IO	O	I	IO	O	I	IO	O
First Open	DRW	I	N	N	N	N	N	N	N	N	N	N	N	N
		IO	N	N	N	N	N	N	N	N	N	N	N	N
		O	N	N	N	N	N	N	N	N	N	N	N	N
	DW	I	N	N	N	Y	N	N	N	N	N	Y	N	N
		IO	N	N	N	N	N	N	N	N	N	Y	N	N
		O	N	N	N	N	N	N	Y	N	N	Y	N	N
	DR	I	N	N	N	N	N	Y	N	N	N	N	N	Y
		IO	N	N	N	N	N	N	N	N	N	N	N	Y
		O	N	N	N	N	N	N	N	N	Y	N	N	Y
	All	I	N	N	N	Y	Y	Y	N	N	N	Y	Y	Y
		IO	N	N	N	N	N	N	N	N	N	Y	Y	Y
		O	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y

I = input only
O = output only
IO = input/output
Y = 2nd, 3rd, . . . open is allowed
N = 2nd, 3rd, . . . open is denied

Figure 4. File Sharing Matrix

(Editor's note: The following is reprinted from the IBM PC Network SNA 3270 Emulation Availability Notice (6322526).)

System Requirements
There are four IBM PC Network Program system configurations that can be started on the IBM PC Network. The minimum requirements vary by configuration. The network supports the IBM Personal Computer AT, IBM Personal Computer XT, IBM Personal Computer or IBM Portable Personal Computer with an 80 column display configured as one of the following:

Redirector: Requires a minimum of 128KB of memory and one double-sided diskette drive.

Receiver: Requires a minimum of 192KB of memory and one double-sided diskette drive.

Messenger: Requires a minimum of 256KB of memory and one double-sided diskette drive.

Server: Requires a minimum of 320KB of memory, one double-sided diskette drive and one fixed disk.

These memory configurations make at least 64KB of memory available for user programs or DOS external commands. An IBM PC Network Adapter, the IBM PC Network Program and the Disk Operating System (DOS) 3.10 are required for each workstation. An IBM-compatible printer is required for print server functions.

IBM PC Network Program Application Compatibility

IBM Personal Computer program license agreements permit the use of a program on a single machine. The customer is responsible for ensuring that each user on the network is appropriately licensed to use any program shared over the network.

The following programs have been tested and are supported to run on the IBM PC Network Program using the IBM PC Network Adapter and DOS 3.10. Please see the IBM PC Network Program User's Guide for installation instructions and application limitations.

Note: The column headings on the following charts are defined below.

- **Share Application** tells you if you can share the application over the network.

  Yes (one): You can share the application, and one network user can use the application at a time.

  Yes (Multiple): You can share the application and several network users can use the application at the same time.

  No: You cannot share the application. Each network user must have a copy of the application to use at his or her computer.

- **Access Data Remotely** tells you about reading and writing data files for the application over the network.

  Yes: You can share (read and write) data for the application across the network.

  No: You cannot share (read and write) data for the application across the network or that the application doesn't allow you to store data on a fixed disk.

Application and Version	**Share Application From Server**	**Access Data Remotely**
APL Version 1.00	Yes (One)	Yes
Application Display Management System Version 1.00	Yes (Multiple)	Yes
Asynchronous Communications Support Version 2.00	Yes (Multiple)	Yes
BASIC Compiler Version 1.00	Yes (Multiple)	Yes
BASIC Interpreter Version 3.10	Yes (Multiple)	Yes
BASIC Programming Development System Version 1.00	Yes (Multiple)	Yes
BASIC Programming Development System Version 1.05	Yes (Multiple)	Yes
Binary Synchronous 3270 Emulation Version 1.00	Yes (One)	Yes
COBOL Compiler Version 1.00	Yes (Multiple)	Yes
Computers and Communications Version 1.00	N/A	N/A
Data Encoder Version 1.00	Yes (Multiple)	Yes
Diskette Librarian Version 1.00	Yes (One)	Yes
Display Communications Version 1.10	Yes (One)	Yes
DisplayWrite 1 Version 1.00	Yes (Multiple)	Yes
DisplayWrite 2 Version 1.10	Yes (One)	Yes
DisplayWrite 3 Version 1.00	Yes (Multiple)	Yes
DisplayWrite Legal Support Version 1.00	Yes (Multiple)	Yes
DisplayWrite Medical Support Version 1.00	Yes (Multiple)	Yes
Dow Jones Reporter Version 1.00	No	Yes
DOS Version 3.10	Yes (Multiple)	Yes
EasyWriter Version 1.15	Yes (Multiple)	Yes
Exploring the IBM PC Network Version 1.00	Yes (Multiple)	N/A

Figure 5A. IBM PC Network Program Application Compatibility

Application and Version	Share Application From Server	Access Data Remotely
FileCommand Version 1.00	Yes (Multiple)	Yes
Fixed Disk Organizer Version 1.00	No	Yes
FORTRAN Compiler Version 1.00	Yes (Multiple)	Yes
FORTRAN Compiler Version 2.00	Yes (Multiple)	Yes
Graphical Kernel System Version 1.00	Yes (Multiple)	Yes
Graphics Development Toolkit Version 1.00	Yes (Multiple)	Yes
Graphics File System Version 1.00	Yes (Multiple)	Yes
Graphics Terminal Emulator Version 1.00	Yes (Multiple)	Yes
IBM Filing Assistant Version 1.00	Yes (Multiple)	Yes
IBM Graphing Assistant Version 1.00	Yes (Multiple)	Yes
IBM Planning Assistant Version 1.00	Yes (One)	Yes
IBM Reporting Assistant Version 1.00	Yes (Multiple)	Yes
IBM Writing Assistant Version 1.00	Yes (One)	Yes
IBM 3101 Emulation Version 1.00	Yes (Multiple)	Yes
Learning to Program in BASIC Version 1.00	N/A	N/A
Learning DOS 2.0 Version 1.00	N/A	N/A
Link (DOS 3.1)	Yes (Multiple)	Yes
Logo Version 1.00	Yes (Multiple)	Yes
Macro Assembler Version 1.00	Yes (Multiple)	Yes
Macro Assembler Version 2.00	Yes (Multiple)	Yes
Multiplan Version 1.10	No	Yes
Multiplication Tables Version 1.00	N/A	N/A
Office Correspondence Retrieval System (OCRS) Version 1.00	Yes (One)	Yes

Figure 5B. IBM PC Network Program Application Compatibility

Application and Version	Share Application From Server	Access Data Remotely
Pascal Compiler Version 1.00	Yes (Multiple)	Yes
Pascal Compiler Version 2.00	Yes (Multiple)	Yes
PC Network Program Version 1.00	Yes (Multiple)	Yes
PC Network SNA 3270 Emulation Program Version 1.00	No	Yes
PCWriter Version 1.00	Yes (Multiple)	Yes
PCWriter Version 1.10	Yes (Multiple)	Yes
PeachText Version 1.00	Yes (Multiple)	Yes
Personal Communications Manager Version 1.00	Yes (Multiple)	Yes
Personal Editor Version 1.00	Yes (Multiple)	Yes
pfs:file Version 1.05	No	Yes
pfs:report Version 1.05	No	Yes
Plotting System Version 1.00	Yes (Multiple)	Yes
Private Tutor Version 2.00	Yes (Multiple)	Yes
Professional Editor Version 1.00	Yes (Multiple)	Yes
Professional FORTRAN Compiler Version 1.00	Yes (Multiple)	Yes
SCRIPT/PC Version 1.00	Yes (Multiple)	Yes
SNA 3270 Emulation and RJE Support Version 1.00	Yes (Multiple)	Yes
Solutions Series: IBM Accounting Solutions Version 1.00	N/A	N/A
Solutions Series: IBM Executive Solutions Version 1.00	N/A	N/A
Solutions Series: IBM Home Solutions Version 1.00	N/A	N/A
Sort Version 1.00	Yes (Multiple)	Yes
Teacher's Quiz Designer Version 1.00	Yes (One)	Yes
VisiCalc Version 1.20	No	Yes
WordProof Version 1.00	Yes (Multiple)	Yes

Figure 5C. IBM PC Network Program Application Compatibility

What's New in DOS 3.10

John Warnock
IBM Corporation

IBM Personal Computer Disk Operating System (DOS) version 3.10 is an enhancement of DOS 3.00. DOS 3.00 supports the IBM Personal Computer AT and includes features for improved file handling and multiple country support.

DOS 3.10 contains all the features of DOS 3.00 plus support for the IBM Personal Computer Network. Also, the DOS Reference manual and DOS Technical Reference manual have been updated and improved. This article details some of the new features and improvements.

Commands and Features

Some of the existing commands and features in DOS 3.00 have been enhanced for version 3.10.

The DOS Linker program (LINK.EXE) has two new parameters. At runtime, the /X parameter lets you adjust the maximum number of segments that the .EXE file contains. You may use any value between 0 (zero) and 1024; 256 segments is the default. The /O parameter allows the 2.30 version of the Linker to work with object modules created by the 1.00 versions of the Pascal and FORTRAN compilers.

The LABEL command has been improved in version 3.10. It now displays the volume label you are working with. When deleting a label, you are now prompted to confirm your choice.

The TREE command now displays files from the root directory when the /F parameter is used. Previously, only files from subdirectories were displayed.

In DOS 3.10, two new commands give you greater control over files and subdirectories.

The JOIN command lets you join the directory of another disk to an empty subdirectory of your current disk. The format is:

```
[d:][path] JOIN [d: d:\directory] or

[d:][path] JOIN [d:/D]
```

where

[d:][path] is the drive and path where the JOIN command file is located

d: specifies the drive to be joined

d:\directory specifies the drive and directory to be joined to

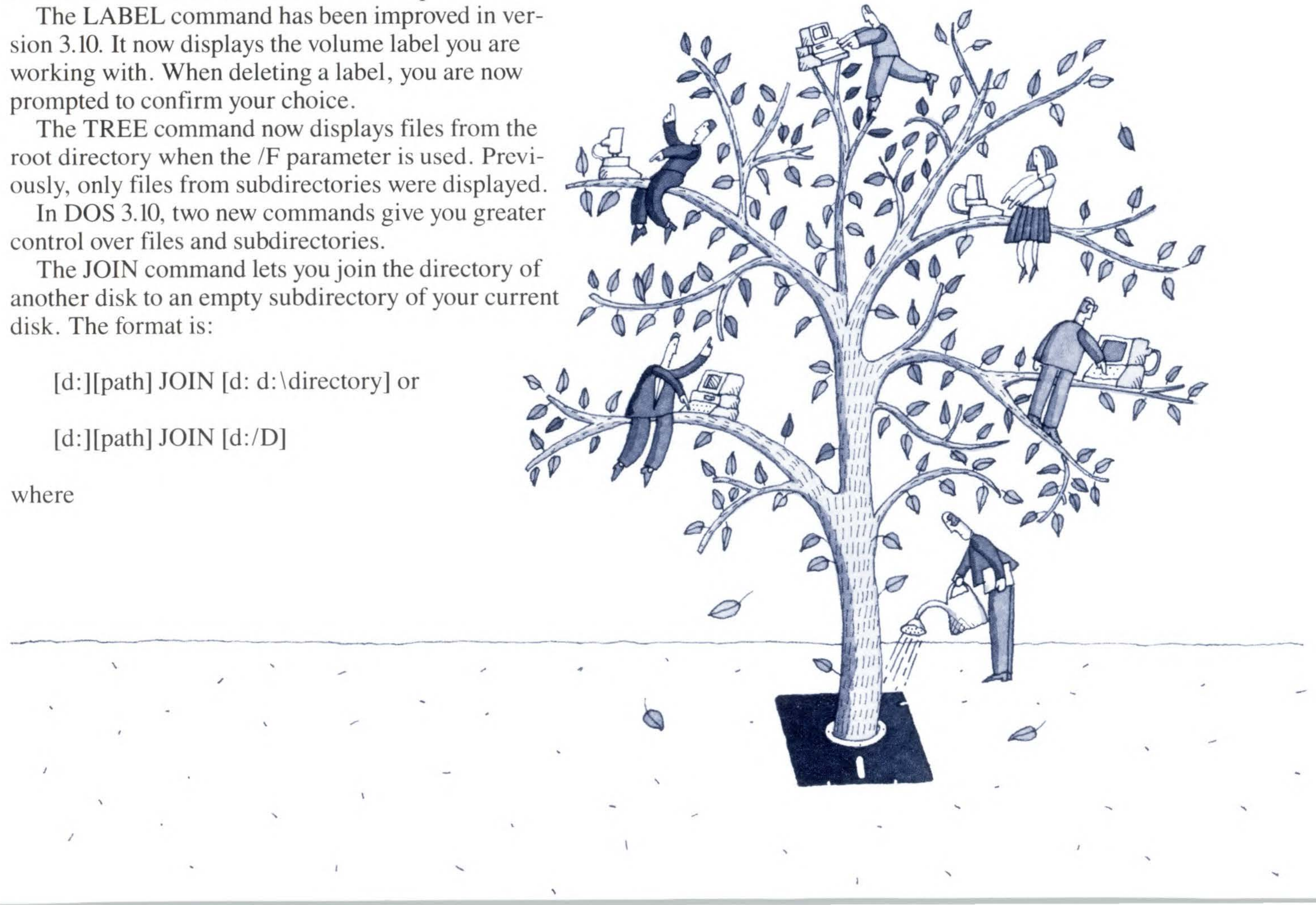
d:/D specifies the drive to be disconnected from a join

You cannot join to a directory that isn't empty, nor can you join to a directory you are currently in. Once a drive is joined, you must disconnect it to refer to the drive by itself. The JOIN command without any parameters shows currently joined drives and directories.

The SUBST command is now the preferred method for making drive reassignments. Like the ASSIGN command, SUBST lets you substitute one drive for another. SUBST lets you select a path as well as a drive for substitution. The format is:

```
[d:][path] SUBST [d: d:path] or

[d:][path] SUBST [d:/D]
```



where

[d:][path] is the drive and path where the SUBST command file can be found

d: is the drive letter you will use for a nickname

d:path is the drive and path you will substitute for the nickname

d:/D is the nickname substitution to delete

You cannot use the same values for both drives, nor can you use the default drive for the first drive letter. If you use the command with no parameters, the current substitutions are displayed.

You cannot use substitution with network drives, and should not use substitution with the ASSIGN, BACKUP, DISKCOMP, DISKCOPY, FDISK, FORMAT, JOIN, LABEL or RESTORE commands. You can use the CHDIR, MKDIR, RMDIR and PATH commands with caution.

The SUBST command is affected by whatever you use for the LASTDRIVE command in your CONFIG.SYS file. The LASTDRIVE command lets you set the maximum number of drives your system will handle. The default is E. If you are not using the CONFIG.SYS file or the LASTDRIVE command, then the only values you can use for the drive are A through F.

Putting JOIN and SUBST to Work

JOIN lets you combine several drives as subdirectories of another drive. You can make several smaller disk drives appear as one large device with many subdirectories. This is very useful with the DIR and TREE commands, for instance.

To join your A and B drives to your C drive, type:

```
C>JOIN A: \ADRIVE
```

This creates an empty directory called ADRIVE on your C disk. Next, type:

```
C>JOIN B: \BDRIVE
```

which creates an empty directory called BDRIVE on your C disk. These empty directories point to the physical devices.

To confirm what you have done, type:

```
C>JOIN
```

Your computer displays:

```
A: => C:\ADRIVE
B: => C:\BDRIVE
```

The DIR command shows these new subdirectories. The TREE command with the /F parameter displays the contents of all files on drives A, B and C. JOIN can be used with virtual disks as well. However, certain DOS commands such as CHKDSK will not go past a JOIN. You also should remember that the newly created subdirectories remain on your disk until you use RMDIR to remove them.

The SUBST command can be used to make older applications work with subdirectories. For example, the IBM Personal Editor can start from any subdirectory and can work with multiple disk drives, but it cannot access other subdirectories. SUBST lets you give complicated drive and path combinations simple single-letter nicknames that your application can accept.

For example, say you start Personal Editor from a subdirectory called PE, and you want to access files in your root directory and in a directory called NOTES. Before starting Personal Editor, type:

```
C>SUBST D: C:\
```

to assign your root directory to D, and

```
C>SUBST E: C:\NOTES
```

to assign the subdirectory NOTES to the nickname E. To confirm what you have done, type:

```
C>SUBST
```

The computer displays:

```
D: => C:\
E: => C:\NOTES
```

Now you can start Personal Editor from its own subdirectory and access other directories using the nicknames D: and E:.

The DOS Technical Reference Manual has been reorganized. Many chapters have been expanded and others are new.

DOS Reference Manual

Chapter 1 of the DOS Reference manual lists the features that are new in DOS version 3.10. These include the two new commands, JOIN and SUBST; the ability to use most DOS commands on a network disk or printer; improvements to the LABEL and TREE commands; and an enhanced Linker.

Instructions for preparing the fixed disk are more thorough and detailed. There also is greater detail about each of the options available for the FDISK command. Step-by-step instructions make the process easy to follow.

The chapter on disk directories has been rewritten and reorganized. More information is provided about each directory command.

The chapter on DOS commands now explains how DOS commands can be used with the IBM PC Network, and which commands cannot be used. The chapter also includes revised LABEL and TREE commands, and the new JOIN and SUBST commands.

DOS Technical Reference Manual

The DOS Technical Reference manual has been reorganized. Many chapters have been expanded and others are new.

The chapter on installable device drivers has been expanded. It provides more version-specific information, and gives more details about each bit in the device attribute field. The chapter provides more information about installing block and character device drivers. Charts have been revised and new functions are explained.

The section on extended screen and keyboard control includes more examples. Many existing examples have been expanded for clarity.

File management information is in a separate chapter that explains how to use file management function calls. It details File Control Block (FCB) and Handle function calls, and when each should be used. This section also explains the differences between file input/output in ASCII mode and binary mode.

The Technical Reference manual contains version-specific information about DOS interrupts and function calls, and includes new interrupts for networking. These new interrupts include:

5E00H	Get Machine Name Returns a 15-byte string with the name of the local computer. The PC Network Program must be loaded for this function to work.
5E02H	Set Printer Setup Places up to 64 bytes of setup information in front of all files destined for a particular network printer. It requires the IBM PC Network Program.
5E03H	Get Printer Setup Returns the address of the printer setup buffer where the printer setup string can be examined. The PC Network Program is required.
5F02H	Get Redirection List Entry Returns local and network device information for each network device created through a Redirect Device call.
5F03H	Redirect Device Allows local print or disk requests to be buffered and sent to other network devices for handling.
5F04H	Cancel Redirection Cancels previous device redirection.

Interrupt 2FH has been renamed from the specific function of Printer interrupt to the more generic term Multiplex Interrupt. The function and error codes are unchanged. An example of how to use the interrupt is included.

BATch Files for the IBM PC*jr*

Ron Nutter
Lexington IBM PC User Group
Lexington, Kentucky

Although the documentation for the IBM PC*jr* does not offer a detailed explanation of BATch files, they can be very useful if you know how to create and use them.

Below is a listing of the AUTOEXEC.BAT file that I use on my PC*jr*.

```
echo off
cls
verify on
logo
mode c80
mode com1:=4800,n,8,1,p
   >nul:
mode lpt1:=com1:
graphics
ddate
cls
time
fc2/q/rh-
```

In the above file:

ECHO OFF—Without this statement, all commands in the AUTOEXEC file would display on the screen as they are executed. With ECHO OFF, the commands in the file will execute without printing to the screen.

CLS—Clears the screen.

VERIFY ON—Any time the disk is written to, it will be verified for accuracy against what is resident in memory. This is a big help if you are entering an important or long file; you want to be sure that it will be usable later.

LOGO—This file is a title screen that comes up and informs me I am using one of my main system disks. After a few seconds, the bootup procedure continues.

MODE C80—For those of you who have an IBM PC*jr* Color Display or other monitor that can display 80 columns, this command lets you use the full 80 columns. Without this command, the PC*jr* defaults to 40-column mode. The C tells DOS that you are operating a color monitor. If you use a separate diskette for system bootup, or wish to set up your program diskettes to use the full 80 columns, you need to copy the MODE.COM file to your program diskettes.

MODE COM1:4800,N,8,1,P >NUL:—I use a serial printer on my system, connected via the serial adapter cable. The serial port is now initialized for 4800 b.p.s., no parity, eight data bits and one stop bit. The P instructs the computer to continually retry the printer in case of a device timeout.

In simpler terms, if you are using a serial printer with an internal buffer, or your printer is slow, you will occasionally get a device timeout message on the screen. The P option stops this from happening. If you were to type this in at the A> prompt, DOS would echo back with COM1:4800,N,8,1,P. Since this is one of a series of multiple statements in a BATch file, >NUL: redirects the echoed statement to a null or nonexistent device. If you have the PC*jr* Internal Modem installed, the serial port for your printer is COM2 unless your hardware was modified when the modem was installed.

MODE LPT1:=COM1:—Some software expects to find a parallel printer connected to the computer and is referred to as LPT1. This statement tells DOS to redirect all output addressed to LPT1 (line printer #1) to COM1 (communications port #1).

GRAPHICS—This COM file becomes an extension of DOS and lets you dump graphic screen contents to the printer when you press the Shift and PrtSc keys.

DDATE—This COM file is a handy utility available in many club libraries. It remembers the last date you entered into the system and lets you make corrections. It is nice to have if you don't have a battery backup clock on your system, or in place of the regular DOS DATE command.

TIME—This prompts DOS to ask you for the current time in military designation, e.g., 1:30 p.m. is 13:30.

FC2/Q/RH-—FileCommand (FC) from IBM is a very useful utility that occupies very little space in memory. /Q instructs FC to execute immediately. RH- tells the program to read the disk only when instructed instead of every time you press Enter. (FileCommand is available from IBM through The Directory, 1-800-IBM-PCSW.)

To set up your AUTOEXEC.BAT file:

1. At the DOS A> prompt, type COPY CON: AUTOEXEC.BAT and press Enter.
2. Proceed to type in, line by line, either the listing above or one of your own. Press the Enter key at the end of each line.
3. After you enter all the lines in your BATch file, press the Fn key and the number 6 key. ^Z will appear on the screen; press Enter again.

The disk drive light will come on, and a few seconds later a message will appear on the screen indicating that a file has been copied. Your BATch file has now been written to disk.

This is one way to create a BATch file. Another way is to use a word processing program that creates ASCII files with no special control characters or file storage schemes.

You'll find that BATch files can simplify and enhance not only your setup procedures, but almost any repetitive task.

One inconvenience of BATch files is that they do not allow you to interact with the computer once the BATch file begins execution. A simple public domain program called ASK.COM allows you to interact with a BATch file while it is running. ASK.COM even allows you to define the prompt or message that appears on the display.

As I explained above, I redirect all output going to LPT1 to the COM1 port. There are times when I would like DOS to set up the system differently. Using the ASK.COM program, I can have the BATch file ask me if I want DOS to configure my system the usual way.

If you don't have a program that allows you to interact with a BATch file while it is executing, you can create a BATch file, CONFIG2.BAT, that has the new configuration. For example, after the DOS A> prompt, type:

```
copy con: config2.bat
mode lpt1:80,6,P  >:nul:
```

and press the F6 key to conclude. This BATch file directs your printer output to the parallel port and initializes the printer for 80 characters per line, six lines per inch.

Then, to get the new configuration, enter CONFIG2 when the AUTOEXEC.BAT file is finished.

Before you try editing your present AUTOEXEC.BAT file to find what works best for you, be sure you save your present AUTOEXEC.BAT file. One way to save the current AUTOEXEC.BAT file is to enter the following after the DOS A> prompt:

```
copy autoexec.bat
   autoexec.bak
```

You'll find that BATch files can simplify and enhance not only your setup procedures, but almost any repetitive task.

Adding an End-of-File Marker

Sig Rosenthal
Westchester PC Users Group
Yorktown Heights, New York

I recently created a file using the redirect feature of DOS 3.00. When I tried to read it into my word processor, I got the following message:

```
Read error or not EOF
```

If I then gave a command to the word processor, I got the dreaded:

```
PARITY ERROR
```

which necessitated a power-off/power-on cycle. I also have had this problem with files created by:

```
COPY CON: filename
```

This time I felt like doing a little exploring. Using DEBUG, I discovered that most of my files ended with a hex "1A" character. When I inserted the "1A" at the end of my file, all the problems went away and I was able to use the file normally.

For anyone who wants to fix a similar problem, here's how: Start DEBUG for the file you want to fix. You must type the Enter key after each command.

```
B: >DEBUG test.doc
```

At the DEBUG prompt symbol "-", type in d. This will cause DEBUG to dump the file. By looking at the left side of the dump, you can see where in the file you are. Keep typing in d until you find the end of your file.

The TEST.DOC file I will use for illustration contains the following two lines:

> This the first line of a test file.
> This is the last line of the test file.

You can find these two lines in Figure 1.

```
-d

 1E26:0100  54 68 69 73 20 74 68 65-20 66 69 72 73 74 20 6C   This the first l
 1E26:0110  69 6E 65 20 6F 66 20 61-20 74 65 73 74 20 66 69   ine of a test fi
 1E26:0120  6C 65 2E 0D 0A 54 68 69-73 20 69 73 20 74 68 65   le...This is the
 1E26:0130  20 6C 61 73 74 20 6C 69-6E 65 20 6F 66 20 74 68   last line of th
 1E26:0140  65 20 74 65 73 74 20 66-69 6C 65 2E 0D 0A 5D C3   e test file...]C
 1E26:0150  E9 18 01 08 46 16 8D 3E-84 3E 8B 46 0E E8 17 01   i...F..>.>.F.h..
 1E26:0160  72 03 E9 06 01 08 46 16-8D 3E 90 3E 8B 46 1E E8   r.i...F..>.>.F.h
 1E26:0170  05 01 72 03 E9 F4 00 26-89 47 08 1F F6 46 0B 80   ..r.it.&.G..vF..
```

Figure 1.

Notice that each line in the file ends with a carriage-return ("0D") and line feed ("0A"). This is usually a good way to locate the end of the file. As a check, the length of the file is contained in the CX register. The length has to be incremented by one to make room for the end-of-file (EOF) character "1A".

Display the CX register:

```
-rcx
CS 004E
:
```

DEBUG displays the contents of the CX register (in this case, "4E"). Add one to the number shown, and type in your response after the colon. Remember, you're adding hexadecimal.

```
:4f
```

This makes the length of the file one character longer than it was. Now we have to add the one character that we made room for, by entering the EOF character, hex "1A", into the end of the file.

Because the file starts in location 100 hex (not 00), you must add 100 hex to the number that was in the CX register to find the actual location of the end of the file. The CX register originally contained 4E. (For long files I use the CX register to find the end of the file, rather than continually entering the "d" command.)

In the dump, each column is identified by a hexadecimal number. The first column is "0", the second is "1", and so on. The eleventh column is "A" (remember we are counting in hex) and the last is "F".

Now, referring to Figure 1, enter the EOF character in the row :0140 in the column E. Tell DEBUG to enter a character into position 14E in the file:

```
-e 14e
```

DEBUG responds with:

```
2E7A:014E 5D.
```

which means that 14E currently contains a 5D (verify this in Figure 1). Now type in the new character:

```
2E7A:014E 5D.1a
```

Note that I did not have to use uppercase letters.

Now you need only write the file out to disk(ette) and quit DEBUG:

```
-w Writing 004F bytes

-q
```

TopView Programmer's ToolKit

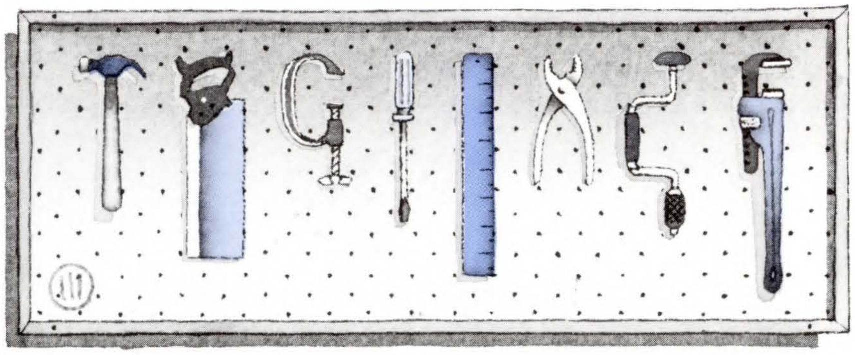
Michael Engelberg
IBM Corporation

The TopView Programmer's ToolKit provides the system interfaces, programming tools and guidelines for developing applications that will take advantage of the TopView operating environment.

The interfaces and tools are intended to make it easier to write programs that will run with TopView. These tools and interfaces produce system code that you can use in your application program and that TopView can understand. For example, if you want your program to do such things as windowing, multitasking and data transfer, you should use the interfaces and adhere to the guidelines in the Programmer's ToolKit.

The Programmer's ToolKit includes a Window Design Aid for developing windows, menus, help screens, error screens and forms to be used within an application.

The ToolKit also includes language interfaces to the IBM Macro Assembler and Pascal Compiler; utilities for merging panel files and converting panels to linkable object modules; and interfaces for writing your own pointing device driver.



Window Design Aid

The Window Design Aid is an interactive tool that allows you to design, construct, store and maintain panels in the TopView environment.

A panel is a stream of data. That data stream creates a window. All the information about your windows, such as display attributes, text, fields and data format, is kept in panel files. When your foreground application program displays a window, the program actually calls for a panel file and gives that file to TopView. TopView then builds the window using the information in the panel file.

The Window Design Aid consists of a full-screen editor and several menus for creating panels. Using these menus you can select three kinds of functions. The Edit functions let you enter, move and copy text, lines and boxes. The Panel functions let you select the window border, title, placement and display attributes (color, reverse video, etc.). The Field functions let you define fields within your panels.

Edit Menu

When the Window Design Aid comes up on the screen, you will see the Edit Menu. From this menu you can select and perform several edit functions for creating panels: Draw, Copy, Move, Auto Copy, Erase, Fill Text and Fill Attribute. (Each function is explained below.) The Edit Menu also gives you access to the Panel Menu and the Field Menu.

In most windows you design, you'll probably want to type some text before using the edit functions. To remove the Edit Menu, press button 2 while the pointer is within the menu. You can redisplay the Edit Menu by pressing button 2 on your pointing device.

The edit functions are:

- **Draw:** The Draw option lets you create lines and boxes in your panel. You can draw single lines or double lines. If you draw boxes, they can have either single-line or double-line borders. You can draw through text or around text, or you can draw lines or boxes and add text later.
- **Copy:** The Copy option lets you duplicate a desired portion of text. You can duplicate in as many places on the screen as you desire. You must first mark the area around the text that you want to duplicate. The text that you copy remains in its original place.

- **Move:** The Move option lets you relocate a desired portion of text. You must first mark the text you want to relocate, then select the Move option, and finally mark the new location. The text that you move does not remain in its original place.
- **Auto Copy:** The Auto Copy option lets you automatically duplicate a marked area wherever you move the pointer in the window. You can use Auto Copy to duplicate characters, lines or boxes.
- **Erase:** The Erase option lets you erase a marked area of text.
- **Fill Text:** The Fill Text option lets you fill a marked area with any character you specify.
- **Fill Attribute:** The Fill Attribute option lets you fill a marked area with any display attribute you specify. Depending on the type of attributes you are using, you can choose from logical, monochrome or color attributes. For example, if you are using a color monitor, you specify both foreground and background color attributes.

Panel Menu

The Panel Menu offers you a choice of reading or writing a panel, an overlay or a template (explained below). It also gives you access to a Panel Options Menu, where you can define your window's features and set its attributes.

The Panel Options Menu lets you specify your window's border, title, position (top or bottom), attributes and display type.

The panel options are:

- **Window Border:** A window border is the box around a window. You can turn the window border on or off.
- **Window Title:** If a window border appears, the window title is shown inside the upper left corner of the border.
- **Window Position:** The window position determines how your window will appear when your application is running. If you select Window on Top, the window becomes the topmost visible window. If you select Window Hidden, the window is initially hidden. You would use Window Hidden if your application must move or size the window before making it visible.
- **Window Attributes:** Window attributes define the logical and physical display attributes for your window.

  A physical attribute is passed unchanged to the display adapter currently in use; if you change from monochrome to color display, or vice-versa, the physical attribute you specified for one display may be incorrect for the other display. Consequently the concept of logical attributes was developed.

  A window in TopView can use 16 logical attributes. Each logical attribute is assigned both a monochrome and color physical attribute. Eight of these attributes are defined by TopView; the other eight can be modified by the application.

  To be more specific, each window in TopView is defined by a logical attribute table that contains eight logical attribute settings for monochrome displays and another eight logical attribute settings for color displays. Each monochrome logical attribute setting is a pointer to the monochrome attribute table that contains actual hardware device codes. Likewise, each color logical attribute setting is a pointer to the color attribute table that contains actual device codes.

  The advantage of using logical attributes rather than physical attributes is that the attributes are automatically chosen to suit the display currently in use. If you have selected Logical Attributes in the Panel Options menu, you can then make changes to the Logical Attribute table by invoking the Set Attribute option in the Panel Options Menu.
- **Display Type:** The display type allows you to specify whether your panel is to be used with a monochrome or color display when you are using physical attributes.
- **Standard Attributes:** The Uses Standard Attrs option allows you to specify that your program uses the TopView standard attributes. Using standard attributes allows the TopView user to change the colors for the logical attributes. Otherwise, the attributes you specify will remain in effect.

In addition to all the panel options you can access from the Panel Menu, you also can use the Panel Menu to write and read panels, overlays and templates.

Both a panel and an overlay contain information describing the appearance of a window. The difference between a panel and an overlay is that a panel supplies the initial information for creating a new window and setting its size, screen location and fields, whereas an overlay replaces the contents of an existing window and may change its size, screen location and fields.

A template is a portion of a panel that has text, attributes and a particular position on the screen. For example, if you create a window that has a business logo in the bottom right corner, and you want to duplicate that logo on every window in the application, you can store the logo in a template file. Then, whenever you read in that template file and write it to a window, the logo is placed in the same location with the same attributes. You normally mark the beginning and ending points of the template area before writing a template to a file; if you do not mark an area, the entire screen is written as the template.

You can write or read a panel, overlay or template by specifying a file name and extension. You can optionally specify a drive and path.

Field Menu

The Field Menu lets you define, copy and move fields in your panel. You can define fields as input, output, select or inactive. You can determine how data in the fields will look; whether you want to see field numbers; and whether you want to see all the data a user enters or just new data. All of these functions are explained in the following text.

The functions in the Field Menu are:

- **Field Options:** When you select Field Options, the Field Options Menu appears. Using the Field Options Menu you can select how your program receives the information a user supplies.

  The functions in the Field Options Menu are:

  —*Field Format:* If you select Modified Fields Only, your program will receive only those fields that the user has changed. If you select All Fields, your program receives every field in your panel, whether or not the field has changed. You should select No Data only if your program reads the window directly.

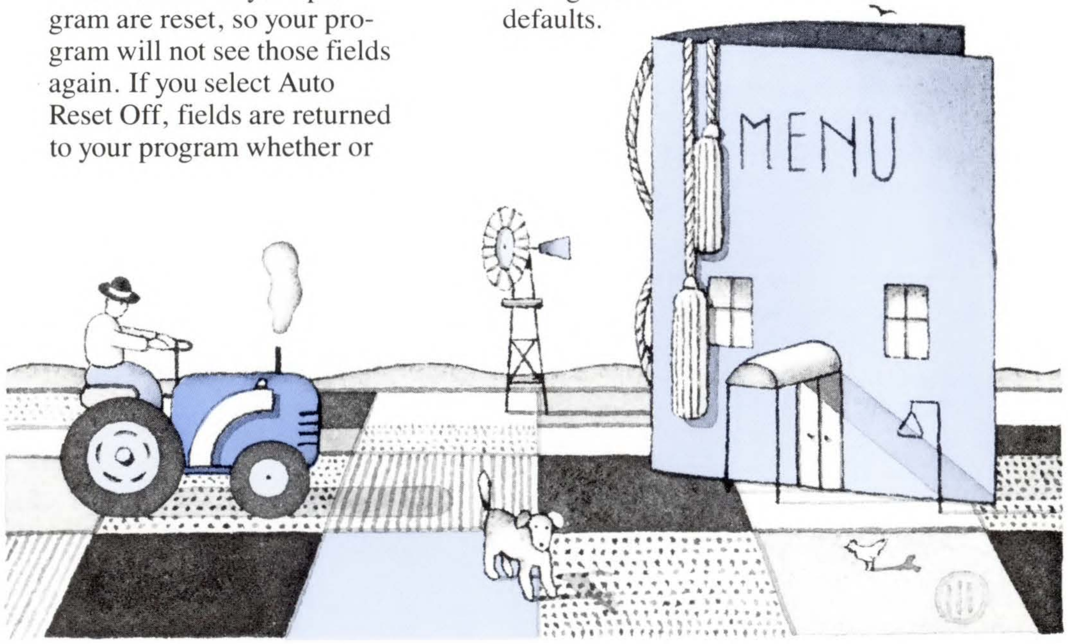
  —*Field Numbers:* Whenever you define a field, it is given a number. If you select the Field Numbers option, your program will see the field numbers associated with the fields in your panel. If you select No Field Numbers, your program will not see the field numbers.

  —*Auto Reset:* Auto Reset controls which fields are returned to your program. If you select Auto Reset On, any fields that were returned to your program are reset, so your program will not see those fields again. If you select Auto Reset Off, fields are returned to your program whether or not they have already been returned.

  —*Select Field Format options:* Select fields are those that the user selects from menus, lists, tables, etc. You can choose how you want select field information to return to your program.

  If you choose Select Field Code, you will receive a Boolean result—Y if the field was selected, N if not. This format is useful for determining which menu choice the user has selected. If you choose Select Field Data you will receive the actual text from the field. This format should be used when the application changes the contents of fields, such as a directory list.

  —*Button Usage options:* Normally button 1 is used to select a field (which may bring up another window), and button 2 is used to end a window and return to the previous status. You can change these settings in the Field Options Menu, although the recommended settings are the defaults.



—*Pointed At Attribute:* When the pointer is in a particular field in a menu, that field is being pointed at. Each pointed at field has an attribute associated with it. For example, in the Window Design Aid the default attributes of a pointed at field on a color display are light gray foreground and blue background.

When you select Pointed At Attribute you can choose how the field will look when it is being pointed at by the user.

—*Selected Attribute:* When you actually select a field, that field takes on a different set of attributes. In the Window Design Aid the attributes of a selected field on a color display are blue foreground and cyan background.

These attributes also are assigned to default selections, such as in the Panel Options Menu, where TopView gives you a predetermined selection.

- **Define Fields:** When you select Define Fields, the Define Fields Menu appears. The Define Fields Menu lets you specify the kinds of fields you want in your panel. You can also move, copy and adjust the sizes of fields. You define a field the same way you mark an area, by using button 1 to mark the beginning and ending points.

  Functions in the Define Fields Menu are:

  —*Field Number:* The system assigns a field number to each field you define. Field numbers are assigned in numeric order. You can reassign numbers if you wish.

  —*Field Type:* There are four field types—Input, Output, Select and Inactive.

  Input: Input allows both the programs and the user to enter data into a field. The data entered will replace any data already in the field.

  Output: Output allows only the program to write data into a field.

  Select: Select allows the user to make a choice by using a pointing device. An application can write data into a select field.

  Inactive: An inactive field merely holds an inactive place in the field table describing your panel. If you have previously defined a field and now want to delete it, you can make it inactive, eliminating the need to renumber all fields.

  —*Copy Field:* You can make one or more copies of a field within a panel by using the Copy Field option.

  —*Move Field:* You can move a field anywhere within a panel by using the Move Field option.

  —*Adjust Field Size:* You can change the size of a field in your panel by using the Adjust Field Size option.

  —*Program Output:* The Program Output attribute enables a group of fields to be written to in one operation at application runtime. If you select No Program Output, the field is not written to during the group output operation.

- **Test Fields:** The Test Fields function lets you simulate using the panel you are creating. You can read in a panel and enter data in its fields as though you were the user of your program. The Test Fields function immediately shows you how the fields reacted, and it displays attributes of the fields, such as length and starting and ending bytes.

  Results are displayed in a window called the Field Test Input Data window. This window also contains three menu options: Reset Fields, Resume Test and End Test. You must select End Test in order to quit the field test mode and return to the Field Menu.

- **Show or Hide Field Numbers:** You can either show or hide field numbers while using the Window Design Aid. If you want to see the assigned field number when you display the field, you should select Show Field Numbers; if not, select Hide Field Numbers.

- **Renumber Fields:** The Renumber Fields function lets you eliminate all reference to inactive fields by renumbring the remaining active fields in numerical order.

- **Delete All Fields:** The Delete All Fields function gives you a way to erase every field in your panel.

Using Windows When You Write a Program

When you write your program, you can use the windows you designed in the Window Design Aid by writing interrupt calls to TopView in your code. These calls tell TopView to perform the input and output that is necessary to read in and manage the displaying of your windows. The way you specify the call depends on the language you are using.

Panel Utilities

The TopView Programmer's ToolKit contains two utilities that support the use of panels in TopView applications.

The PANELLIB utility merges a group of panels into one file. This simplifies panel operations and reduces the storage required by panels in the application.

The PANELOBJ utility converts a panel or a file of merged panels into an object module. This module can be linked with an application so that the panel resides in storage while the application is executing. This is advantageous in an application in which execution speed is more important than the amount of memory required. The utility also makes it possible for an application to use panels without having to keep panel files on-line.

Creating Program Information

When you write an application for TopView, you should create a program information file for your application. This information tells TopView certain things about your application — its name, where to find the program and its data, memory requirements, and so on.

You should give your program information file name the extension .PIF (with any filename you wish) and place the file on the same diskette or on the same drive and directory as your program file.

In addition to listing the existing IBM software applications that run in the TopView environment, the Add-a-Program Menu offers the option of selecting an "Other" application. By selecting Other the user can add your application to his or her TopView environment.

When the user selects Other, TopView asks the user where to find your program file. After the user tells TopView, TopView goes to that drive and directory and shows the user a list of the names of all files that have the extension .PIF (except TV.PIF). It then appends all .PIF files selected by the user to the Consolidated Program Information File, TV.PIF.

The TopView Programmer's ToolKit includes a utility program to help you create a program information file for your application. This program is called Create Program Information. To add this utility program to your TopView environment, enter the Add-a-Program Menu, select Other, and tell TopView that the program can be found on your ToolKit diskette (or drive and directory). After the Create Program Information utility is added to your environment, you can start it using the Start-a-Program Menu.

When you run Create Program Information, a screen similar to the Change Program Information screen appears. You should fill in your program's information, press Enter and select Save. This creates your program information file with the filename you furnish plus the extension .PIF.

Following are the explanations of the fields on the Create Program Information screen.

- **Program Title:** This is the name by which the user knows the application. This name is displayed in the Start-a-Program Menu and the Switch Menu. It can be up to 30 characters long. The name need not be the same as the name on the application diskette; the user can change it using the Change Program Information function.
- **Program Pathname:** This is the full pathname and program name of the file that starts your application program. For example:

  ```
  C:\mydir\myprog.exe
  ```

  The name of the program file should have an extension of .EXE or .COM.
- **Program Parameters:** This field contains the parameters that should be added to the invocation string of your program. If you enter a question mark as the last character of the string, TopView will prompt the user for the program parameters when the application is started.
- **Data Files Location:** This field points to the default drive and directory of your application's data files — in other words, your program will look for its data on the drive and directory you specify. Note: If your program is added to the TopView environment using the Other option in the Add-a-Program Menu, the location of its data files defaults to the drive and directory specified for Program Pathname.
- **Memory:** These fields indicate the amount of memory required to run your application. You must specify minimum, maximum and system memory. System memory is the memory TopView uses to store your program's system control blocks and video buffer. The minimum system memory is 7KB. A program using graphics requires an additional 16KB of system memory.

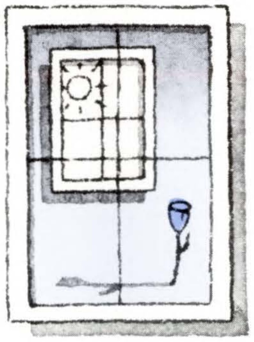
When the user starts your application, TopView tries to provide the maximum amount of memory you specified. If it is available, TopView loads your program into that amount of memory. If the maximum memory is not available, TopView then tries to allocate the minimum memory specified. If the minimum memory is available, TopView can load your program; however, if more than the minimum but less than the maximum is available, TopView allocates the intermediate amount of memory to your program.

Neither the minimum or maximum memory amounts should include memory required for DOS. For example, if a program requires 64KB under DOS, it can run in only 52KB in the TopView environment, because TopView does not use 12KB assigned to DOS.

The maximum that can be specified for a single application is 512KB.

- **Screen Type:** This is a one-character field that indicates which video mode and/or display monitor your application uses. Possible screen types are:

2 =	80 x 25 black-and white
3 =	80 x 25 color
4 =	320 x 200 color
5 =	320 x 200 black-and-white
6 =	640 x 200 black-and-white
7 =	80 x 25 monochrome
X =	Any display can be used, and if your application changes the video mode, TopView will accommodate the change.
D =	Use the current display mode in effect when your application is started. TopView will not accommodate any changes to the video mode that your application makes if the equipment flag bits do not match the specified video mode.



- **Screen Pages:** This is the number of screen pages that your application uses. Most applications use only one page; however, from one to eight pages can be specified.
- **Window Size:** This is the size of the window initially created for your application. It is specified as rows and columns. For most existing applications that use the full screen, the window size is 25 rows by 80 columns.
- **Initial Offsets:** This indicates the initial location at which TopView is to position the window when your application is started. For applications that use the full screen, the offsets are row 0, column 0.
- **Shared Program Pathname:** This is the full pathname of a program that your application can share with other applications. If your application uses the TopView Filter Program (FILTER.EXT) and a filter table, you should specify the filter program's pathname in this field.
- **Shared Program Data:** This is a character string that contains any data that should be passed to the shared program. The string might be a file name or a string of data. If your application uses the TopView Filter Program, you should enter the name of your application's filter table file. A filter table file has a suggested extension of .TBL.
- **Range of Software Interrupt Vectors Swapped:** This denotes the range of software interrupt vectors that your application changes. The range is specified as two fields, a low value and a high value. The maximum range is 00H through FFH. TopView saves this information when another program becomes the foreground application, and restores this information when your application returns to the foreground.
- **Writing to the Screen:** Valid input for this field is Y for yes or N for no. This information is used by TopView to determine if your application writes directly to the video buffer (address B0000H for a monochrome display or B8000H for a color display). If your application writes directly to the video buffer, it cannot run in the background, and TopView Window functions cannot be performed in it.
- **Accessing System Keyboard Buffer:** Valid input is Y or N. Programs that access the BIOS keyboard data areas cannot be allowed to run in the background.

- **Running Only in the Foreground:** Valid input is Y or N. TopView uses this information to determine whether your application should run only in the foreground. If your program writes directly to the video buffer using the TopView INT 10H calls, it is well-behaved, so it can run in the background, and TopView Window functions can be performed in it.
- **Using the Math Co-processor:** Valid input is Y or N. TopView uses this information to determine whether your application uses the IBM 8087 or 80287 Math Co-processor to do arithmetic.

Language Interfaces

Application program should use the TopView system functions to take full advantage of the facilities provided by the TopView environment. The Programmer's ToolKit comes with interfaces between TopView and two application programming languages, assembly language and Pascal. These interfaces allow your program to access the TopView multitasking and windowing functions.

The interface between TopView and an assembly language program is through the Intel 8088 and 80286 software interrupt mechanism.

The interface between TopView and a Pascal program consists of several files that are provided with the Programmer's ToolKit. These files contains libraries of special procedures that access the TopView Assembler routines. A Pascal program should include these interface files by using the metacommand $INCLUDE. The Pascal interface is included to serve as an example of how language interfaces can be written for TopView.

In addition, the Programmer's ToolKit contains sample high-level interfaces for assembly language and Pascal.

You also can write application programs in other languages such as FORTRAN. However, you must write your own interface to the TopView assembly language routines.

Help Screens in the Programmer's ToolKit

Table 1 contains the list of Help screens in the TopView Programmer's ToolKit.

Table 1. Programmer's ToolKit Help Screens

Create Program Information program (3 Help screens)
Window Design Aid program
- Edit Window
  - Window Size/Position
  - Button 1 and Button 2
  - Moving the Pointer
  - Entering Data
  - Marking an Area
  - Function Keys
    - F1 Show Keys
    - F2 Draw
    - F3 Copy
    - F4 Move
    - F5 Auto Copy
    - F6 Erase
    - F7 Fill Text
    - F8 Fill Attributes
    - F9 Vertical Scale
    - F10 Horizontal Scale
    - ESC Status Line
  - Edit Menu
    - Draw
    - Copy
    - Move
    - Auto Copy
    - Erase
    - Fill Text
    - Fill Attributes
    - Panel Menu
      - Panel Options
        - Window Border
        - Window Title
        - Window On Top/ Window Hidden

Table 1. Programmer's ToolKit Help Screens (cont.)

        - Logical Attributes
        - Physical Attributes
        - Display Types
        - Standard Attributes
      - Read Panel
      - Write Panel
      - Read Overlay
      - Write Overlay
      - Read Template
      - Write Template
      - Enter File Name
    - Field Menu
      - Field Options Menu
        - Modified or All Fields
        - No Data
        - Field Numbers
        - Auto Reset
        - Select Field Format
        - Button Usage
        - Pointed-At Attribute
        - Selected Attribute
      - Define Fields
        - Define Fields
        - Field Numbers
        - Field Types
        - Copy Field
        - Move Field
        - Adjust Field Size
        - Input Field Attributes
        - Select Field Attributes
        - Program Output Attribute
      - Test Fields
      - Show/Hide Field Numbers
      - Renumber Fields
      - Delete All Fields

The following Help panels appear only when the user requests help from within the appropriate window; these panels cannot be accessed through the Help menus.

Test Results
Enter Title
Logical Attribute Selections for Fill Attr
Logical Color Attribute Definitions
Logical Monochrome Attribute Definitions
Color Attributes
Monochrome Attributes
Enter Character for Fill Text
Move Fields
Set Border Attribute

The Mechanics of Assembly Language

(Part 3)

David Betts
San Francisco PC Users Group

(Editor's note: The two earlier parts of this series appeared in the March 1985 diskette issue of Exchange *and the June 1985 printed issue of* Exchange.*)*

We started this series by discussing what hexadecimal numbers were. We then continued by covering simple BASE 2 and BASE 16 arithmetic. If you've printed out the little HEX to DEC conversion table using the BASIC program in part 2, we're ready to conclude the series by discussing the basics of silicon arithmetic. (I'll use "8086/8" as shorthand for either of the Intel microprocessors, the 8086 and 8088.)

While every microinstruction in the Intel instruction set for the 8086/8 requires the silicon to do arithmetic, for our purposes we'll consider only 24 "arithmetic" instructions. These instructions are grouped by function in Table 1. A full discussion of each instruction would take many pages and is available elsewhere (see the references at the end of this article). I'll stick to covering some of the basics.

Table 1. 8086/8 Arithmetic Instructions

Addition	AAA, DAA, ADD
Subtraction	AAS, DAS, SUB, SBB
Multiplication	AAM, IMUL, MUL
Division	AAD, IDIV, DIV
Rotate	RCL, RCR
Shift	SAL, SAR, SHL, SHR
Increment	INC
Decrement	DEC
Negate	NEG

The 8086/8 contains four general-purpose registers which are used for most operations and all arithmetic. The names of these registers are accumulator (AX), base (BX), counter (CX) and the data register (DX). As you might infer from the names, each of these registers has special purposes. We will briefly discuss some functions of these registers.

In all four registers, numbers are represented as patterns of 1's and 0's, or binary numbers in the four registers. The size of these binary numbers is arbitrarily limited to 16 bits or digits (a HEX number of 4 digits) by the physical size of these arithmetic registers. This is big enough to represent any (unsigned) integer decimal number from 0 to 65,535, or 2 to the 16th power, or 1111 1111 1111 1111 in binary. Alternately, if we choose to use the topmost place or "bit" to indicate the sign of the number, where we use 1 equal to negative and 0 equal to positive, we can represent any integer between -32,767 to +32,767 inclusive.

In part 2 we discussed the addition and subtraction of binary numbers. You'll recall that binary arithmetic works according to the same mathematical principles as decimal numbers. That is, when you need to, you "carry" and/or "borrow". To see how the microprocessor carries and borrows, let's examine what happens when DEBUG executes some assembly code that adds some numbers.

Start DEBUG. When the prompt comes up, type in the code in Figure 1. In Figure 1, <Enter> means press the Enter key, and the small x's stand for numbers whose values aren't important to us now. DEBUG will convert the instructions into machine code with the (A)ssemble command and then run it with the (G)o command.

```
-R <Enter>
-A 0100 <Enter>
xxxx:0100 MOV AX, FFFF <Enter>
xxxx:0104 MOV BX, 0001 <Enter>
xxxx:0106 ADD AX, BX <Enter>
xxxx:0108 <Enter>
-G 0108 <Enter>
```

Figure 1.

For those of you without DEBUG version 2.x, use the (E)dit command instead of the (A)ssemble command above to enter the code as shown in Figure 2 (pressing the space bar between numbers):

```
-E 0100 <Enter>
xxxx:0100 xx.B8 xx.FF xx.FF xx.BB xx.01 xx.00 xx.01
  xx.D8 <Enter>
```

Figure 2.

The first "R" you typed lists the current contents of the registers. Then, after running the little program you typed in, the registers are automatically displayed again.

Notice particularly the AX and the BX registers. The program's first statement put the HEX number FFFF (DEC 65,535) in register AX, then a 1 in the BX register and ended by adding them and putting the sum into AX. But, if you look at AX, it should presently be all zeros. Why? Because FFFF + 1 becomes 10000, but the register can't hold the top one. To see where it goes, examine another register called the Flag Status register. It's the funny list of letters over on the right side of the register display. For the first register display you should see:

```
NV UP DI PL NZ NA PO NC
```

and for the second register display you should have:

```
NV UP DI PL ZR AC PE CY
```

The last flag (the rightmost pair of letters) shows the "carried" one. This is the carry flag, and it has changed from (N)o (C)arry to (C)arr(Y). If we substituted the instructions to do a subtraction:

```
MOV AX, 0000
MOV BX, 0001
SUB AX, BX
```

(the bytes B8, 00, 00, BB, 01, 00, 29 and D8), we would have seen the Flag Status register change from:

```
NV UP DI PL NZ NA PO NC
```

to:

```
NV UP DI NG NZ AC PE CY
```

where the sign flag (fifth from the left) now says (N)e(G)ative.

By careful programming, we could use the registers and the flags to operate on any size or type of numbers, transferring intermediate results to the other registers or memory. This is how silicon does arithmetic. I'm sure you noticed that other registers were affected, too.

For a more complete picture on what is happening, I suggest studying the chapter on DEBUG in the DOS manual, specifically the table on page 12-50. Other sources of information are listed below.

References:

1. *iAPX 86, 88 Users Manual, Intel Corporation, Literature Department, 3065 Bowers Ave., Santa Clara, CA 95051 (the official information)*
2. *The 8086 Book, Russell Rector and George Alexy, Osborne/ McGraw-Hill, Berkeley, CA 94710*
3. *Assembly Language Programming for the IBM Personal Computer, David J. Bradley, Prentice Hall, Englewood Cliffs, NJ 07632*
4. *8088 Assembler Language Programming: The IBM PC, David Willen, Jeffrey Krantz, Howard W. Sams & Co., 4300 W 62nd St., Indianapolis, IN 46268*
5. *Assembly Language Primer for the IBM PC and XT, Robert Lafore, (The Waite Group), New American Library, 1633 Broadway, New York, NY 10019 (highly recommended for those new to Assembly language)*
6. *Bluebook of Assembly Routines for the IBM PC and XT, Christopher Morgan, (The Waite Group), New American Library, 1633 Broadway, New York, NY 10019*
7. *8086/8088 Microchart, part number ML-8086 (This is not a book, but an 8 x 11 plastic page packed with all the instruction mnemonics, like those chemistry and physics cards you used in college.) Available from Jameco Electronics, Belmont, CA 94002, (415) 592-8097*
8. *Programming the 8086/8088, James W. Coffron, Sybex Inc., 1983*

Legal Protection of PC Software

James Ebetino
IBM Corporation

(Editor's note: The author is an IBM attorney.)

Software developers, whether individuals or business entities, invest time, money and other resources to develop marketable program products. Because software is generally easy to reproduce, software developers who wish to realize a return on their investments rely on contracts, the copyright law and trade secret laws to help them prevent others from misappropriating their products and ideas.

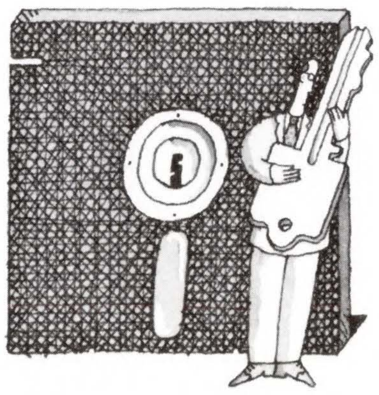
Many software users, on the other hand, do not understand the restrictions and obligations placed on them by these contracts and laws. Inadvertent violation of these contracts and laws can result in civil liability, and willful violations can result in civil and/or criminal liability.

This article is intended to provide software users with a basic understanding of the principles surrounding software usage so that they may recognize certain problem situations and obtain legal guidance when needed.

Contract Protection

Software is generally licensed to users according to the terms of a license agreement. To gain the user's agreement to the licensing terms, the software owner (licensor) may require the user to sign the agreement. Alternately, the licensor may simply notify the user that opening the shrink-wrapped program package, opening the diskette package, or taking some other affirmative act will be construed as agreement to the licensing terms.

Unless the licensor commits fraud or some other unlawful act, the user generally will be bound to the contract terms that he or she has agreed to but may never have read despite the fact that the terms were made available for reading.



License agreement terms vary from licensor to licensor and sometimes from product to product, even if the products are owned by the same licensor. License agreement terms often provide restrictions concerning the copying, transfer, decompiling, modifying, networking and confidentiality of the software. In addition, they often disclaim warranties and limit the liability of the licensor.

Accordingly, before you agree to the licensing terms, it is important that you read and understand the agreements. If you cannot agree to the terms of the license agreement and cannot obtain an amendment or waiver to the objectionable terms (which may often be the case), you should contact the licensor and arrange for a return of the software and a refund of any monies you paid for it. Remember, however, that if you have already agreed to the licensing terms, it may be too late.

As indicated above, licensing terms vary. It is common, however, to see license agreements that are similar in certain areas. For example, many license agreements allow the user to operate the software on only a single computer. Unless the license agreement requires you to specify the serial number of the computer on which the software is to be operated, or otherwise indicates the software is to be used on a particular computer, this restriction should be construed to mean that the software can be used concurrently on only one computer at a time. If this is the case, you can use the software on your computer at work and take it home at the end of the day for use on your home computer. However, you cannot permit it to be used on two computers at the same time unless you have obtained two licenses. Thus, in such a case, networking is prohibited unless all terminals that are being operated concurrently are using separately licensed software.

Similarly, many license agreements permit the user to make a designated number of copies of the software, but only for the purpose of backing up the program. Also, agreements often allow for the transfer of the program, provided that the transferee agrees to the terms and conditions of the license agreement. Finally, most license agreements contain terms relating to copyright or trade secret protection. These forms of protection are discussed next.

Copyright Protection

The United States copyright law helps the owner of a program realize the commercial value of his or her creative efforts. Basically, the law prohibits others from copying (except for making reasonable backup copies), translating, modifying, condensing or expanding the original written work without the permission of the software owner.

The law applies to the written expression of the software. Therefore, the software source and object code, for example, are protected by the copyright law, but the ideas or principles used to develop the software are not protected.

Accordingly, copyrighted software that is distributed without a license agreement still has restrictions associated with its use. When copyrighted software is distributed with a license agreement, the agreement may grant permission to use the software in ways which would not be allowed, without permission, under the copyright laws. However, the license agreement also may create obligations more restrictive than those inherent in the copyright laws.

Trade Secret Protection

State trade secret laws provide the mechanism for software developers to protect unique concepts and ideas they use to develop software. By establishing an obligation of confidence with the user, the software developer can prohibit the user from disclosing the unique concepts and ideas to others and from using the concepts and ideas in other products. Of course, independent development of similar, or even identical, ideas and concepts is not prohibited.

To establish the trade secret relationship, the user must agree to observe the confidentiality of the unique concepts and ideas. Normally this is done through license agreement clauses which state that the program contains "confidential" information or is a "trade secret" of the licensor. Sometimes licensors call the program "proprietary" or restrict the user from "disclosing the program" without indicating that it is a "trade secret" or contains "confidential" information. Whether the latter language creates trade secret obligations is unclear because "proprietary" is an ambiguous term and because nondisclosure obligations may be construed to be contractual in nature rather than establishing a confidentiality obligation. If you are concerned about using trade secret software, you should obtain a clarification of ambiguous language from the licensor.

Trade secret restrictions may be of less concern to individuals or companies who use but do not develop software. For those who do develop software, however, serious complications can arise by agreeing to trade secret obligations with respect to software. For example, if by using the software you have access to the unique concepts and ideas, you must be able to prevent yourself from using those ideas in the software products you develop. This can be extremely difficult if you do not know which ideas in the software are unique and considered confidential, or if the trade secrets you learn are assimilated into your general knowledge in such a way that you no longer can distinguish which ideas are trade secrets and which are not.

If the licensor of the trade secret software can prove that a program you developed uses ideas and concepts similar to the licensor's confidential ideas and concepts and that you had access to the licensor's trade secrets, the burden of presenting evidence of independent development falls on you. As an individual, you may find it extremely difficult to prove independent development of ideas that you were previously exposed to. As a company, it may be possible to prevent certain software developers from being exposed to the ideas, allowing them to independently develop similar software, with procedures in place to prevent them from being contaminated by the trade secrets. However, this can be complex, and proving that the procedures were followed in a given instance can be equally difficult. All in all, the risks associated with software developers using trade secret software must be carefully weighed.

Conclusion

Software users should carefully read and understand the obligations imposed on them before agreeing to licensing terms. Contractual provisions, copyright protection and trade secret protection are all legally enforceable and can subject the user to civil and, in some cases, criminal liability.

Companies that acquire software should establish procedures to insure that all licensing restrictions are acceptable before they are agreed to. This may be particularly difficult without a well-defined and enforced set of software procurement procedures. Depending on the size of a company and the amount of licensed software it uses, it may make sense to centralize all software procurement to make sure that trained personnel see and interpret all license agreements before the company or an employee agrees to the terms. Ambiguous terms should be clarified with the vendor; users should be required to understand and adhere to license agreement restrictions; and, when necessary, appropriate legal guidance should be sought.

Remember, software developers use contracts, copyrights and trade secret protection to enable them to realize a return on the time, money, creative efforts and other resources they invest in software development. Just as with any other products, ownership rights and contract terms governing the use of software should be respected.

Tracking Business Use of Your Home Computer

David Sullivan
Corvallis PC Computer Club
Corvallis, OR

(Editor's note: This article does not discuss income tax aspects.)

The Internal Revenue Service now mandates that, before you can deduct the cost of your home computer, you must have records showing that at least 50% of its use was dedicated to business ventures. This requires you to keep contemporaneous records of each computing session if you intend to deduct your home computer. Although this is a bother, your computer can keep these records for you. Here is a simple way to do it.

Step 1: Type the following BASIC program onto your system disk. Name it IRS.BAS.

```
10 OPEN "IRS.DOC" FOR
   APPEND AS #1
20 CLS
30 PRINT "Enter a description
   of why you turned on the
   computer:"
40 INPUT A$
50 PRINT #1, " On at: " ;
   DATE$; " "; TIME$; "
   Reason: " ; A$
60 CLS : SYSTEM
```

This program prompts you to enter the reason you turned on the computer and appends your reason to the end of a file named IRS.DOC on your disk.

Step 2: Type the next BASIC program onto your system disk. You have to execute it each time you turn off the computer or change from business-related tasks to personal tasks or vice-versa. Name this program OFF.BAS.

```
10 OPEN " IRS.DOC " FOR
   APPEND AS #1
20 PRINT "Enter a description of
   what you did in this session"
30 INPUT A$
40 PRINT #1, "Off at: " ;
   DATE$; " " ; TIME$; "
   Note: "; A$
50 PRINT #1, " "
60 CLS : SYSTEM
```

Step 3: In your AUTOEXEC.BAT file, place the line BASICA IRS.BAS after the DATE and TIME commands, and make sure you have a copy of BASICA.COM where your system can find it. On a two-drive system, have it on the boot disk; on a hard disk system, it can be anywhere if you use the PATH command.

You can type in the command BASICA OFF as the last thing you do before you shut off your computer (or when you change from business items to personal items and vice-versa), or you may want to create a BATch file named OFF.BAT that would contain the command BASICA OFF.

If you find it hard to remember to execute the OFF program, you can put the BASICA OFF command in BATch files that call your programs. For example, to call IBM Personal Editor, you would create a BATch file called PE.BAT. The following lines would be in your PE.BAT file:

```
PE
BASICA OFF
```

To bring up Personal Editor, type PE. When you exit the Personal Editor program, DOS executes the BASICA OFF command automatically. If you use a BATch file to call your programs, be sure your computer has access to the BASICA.COM program, either on each program diskette or somewhere on your fixed disk.

Though I don't generally enter a note when I run the OFF program, I have it there in case I have changed tasks during the session and have not recorded the change, or if I need to leave myself a message about the session.

The IRS is quite specific that it is your responsibility to protect your records from unforeseen events. Always back up your IRS-.DOC file.

Questions about the deductibility of your home computer should be directed to the IRS. They have 1-800-numbers you can call for information.

IBM Personal Computer Travel Information

(Editor's note: This article is adapted from a communication sent to Authorized IBM Personal Computer Dealers.)

It's been said that the IBM Portable Personal Computer makes a perfect traveling companion. It can go where you go and work where you work. And it increases your personal and business productivity wherever you take it—even to the far corners of the world.

In fact, that's true not only of the IBM Portable Personal Computer but of IBM's entire family of personal computers. International airlines have no difficulty transporting any model that meets the dimensional limits for carry-on luggage, and thousands of people take them overseas each year.

But you should be aware that certain conditions can affect the transport, service and performance of IBM Personal Computer hardware and software products when they leave the United States.

U.S. Department of Commerce Regulations

Because these regulations may change from time to time, it's best to get the latest information that will be applicable at the time of the trip. Up-to-date guidance can be obtained by writing the following agency of the U.S. Department of Commerce:

The Office of Export Administration
P. O. Box 273
Washington, DC 20044

U.S. Customs Regulations

Under present regulations, you should have little or no difficulty taking your IBM Personal Computer out of the country. But problems can occur on the return trip. Unless you have the necessary documentation with you, U.S. Customs can impose a duty fee on, hold or confiscate your personal computer at the port of entry.

Be sure to take your sales receipts and invoices to the international airport and file U.S. Customs documents before leaving on a trip.

Service Around the World

The IBM Standard Warranty Agreement is not valid if you take your IBM Personal Computer out of the country. Your machine will not have warranty protection outside the U.S.A.

Despite this, however, help may still be available when you travel.

Be sure to take your sales receipts and invoices to the international airport and file U.S. Customs documents before leaving on a trip.

Authorized dealers in other countries should have the parts and information to service any IBM Personal Computer product if it is available in their country; not all IBM Personal Computer products are available everywhere. Service is largely unavailable for the IBM PC*jr* outside North America, for example, because this product has been announced only in the U.S., Canada and some Caribbean nations. IBM does not support personal computer products in countries where they have not been announced.

Power Supply Requirements

Other countries use line voltages between 100 and 250 volts. Plugging a 110-volt domestic machine into a 220-volt source will instantly damage the machine's power supply.

There are two ways to avoid these voltage problems: install a new power supply appropriate for the country where the personal computer is being used, or use a step-down transformer to compensate for the voltage difference.

The IBM Portable Personal Computer and the IBM Personal Computer AT have universal power supplies that — when coupled with the proper electrical outlet adapter plug — can be used virtually anywhere a U.S. traveler is likely to go.

Software Concerns

You may have trouble when you attempt to take software into other countries. Some foreign customs agencies have strict regulations regarding the importation of diskettes and fixed disks, and some foreign airport security scanners may erase the data written on them.

We Recommend...

IBM recommends that travelers carrying personal computer products out of the country be aware of and comply with government regulations in the area of high-technology transport between the U.S. and other countries.

New Products

Hardware

IBM Color Jetprinter
The IBM Color Jetprinter is a drop-on-demand color ink-jet printer that provides seven-color, all-points-addressable color graphics at 100 X 96 pels-per-inch and 100 X 72 pels-per-inch resolutions, as well as near-letter-quality and draft text. It can be manually changed between two printing intensity modes, normal and bold. The Color Jetprinter has pin feed and cut sheet paper capability, four printing pitches, and super- and sub-script characters. It can print up to 132 characters on an 8-inch line, and create transparencies directly on special transparency forms.

The Color Jetprinter comes with two character sets: PC Character Set 2 and 3270 Extended character set. It prints quietly at 46 decibels while printing 20 characters per second (cps) for near-letter quality or 50 cps for draft quality at 16.7 characters per inch.

For a six-month period ending November 1, 1985, the IBM Color Jetprinter will be shipped with the licensed spooler program Print Buffer at no additional charge.

The Color Jetprinter attaches to all members of the Personal Computer family and the 3270-PC, 3270-PC/G, and 3270-PC/GX through a standard parallel interface.

IBM Proprinter
The IBM Proprinter is a nine-wire matrix impact printer that offers both tractor and friction feeds for easy printing on all forms. The manual front sheet feed handles cut sheet forms up to four parts thick, as well as envelopes. Versatile features of the Proprinter include print speeds of 200 characters per second (cps) in data processing mode, 100 cps in text mode, and 40 cps in near-letter-quality mode; incremental line spacing up to 1/216 inch; six printing pitches; the ability to print downloadable fonts; and graphics printed using 480, 960 or 1,920 dots per 8-inch line.

In the third quarter of 1985, the Proprinter will have two additional options: an Asynchronous Serial Interface module and a 5KB print buffer.

The Proprinter attaches to all members of the Personal Computer family, the IBM Industrial Personal Computer, 3270-PC, 3270-PC/G, 3270-PC/GX and IBM Series/1 System Unit 3170 Model 495 through a standard parallel interface or a user-installable Asynchronous Serial Interface (available in the third quarter of 1985).

IBM PC/Videotex
IBM PC/Videotex allows any member of the IBM Personal Computer family (except the Portable PC) to function as a videotex terminal. The personal computer user may establish communications with a videotex host, receive videotex frames for display on a color monitor or TV, enter data for transmission back to the host, save incoming videotex frames on disk, view the saved copies and display videotex frames generated from local user-written applications.

PC/Videotex supports a subset of the service reference model (SRM) of the North American Presentation Level Protocol Syntax (NAPLPS) standard. It also offers terminal support for the IBM Series/1 Videotex System (SVS/1) in an implementation of the NAPLPS standard.

PC/Videotex connects to a variety of videotex networks using either asynchronous or X.25 protocols.

Videotex frames captured on disk are identified by a user-provided library name that is used to replay the frames. Continuous, sequential replay of captured frames is controlled with a user-specified time delay. Or, the user may select frames from a videotex data base created with service-provided programs (for the host and the IBM Personal Computer) which interface to a utility program distributed with PC/Videotex. The PC/Videotex utility program allows frames and routing information to be copied from DOS files into a frame library, and it provides other library services.

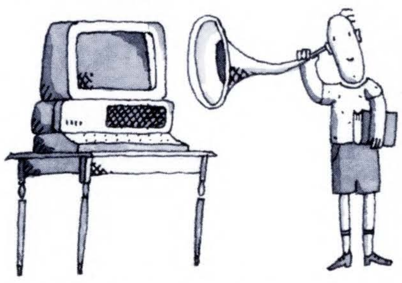
IBM PC/Videotex comes in two versions: Version A1.00 for the IBM PCjr Enhanced Model and Version B1.10 for the IBM Personal Computer, Personal Computer XT and Personal Computer AT.

Version A1.00 requires a PCjr Enhanced Model with DOS 2.10 or higher; 128KB of memory; one double-sided diskette drive; a color display or TV; SMARTMODEM 1200 or 1200B or equivalent, or WD212-X modem (required for X.25 LAPB support) or equivalent, or a 1200-baud manual-dial modem; and appropriate adapter cables for the display and modem.

Version B1.10 requires DOS 2.00 or higher; 128KB of memory; one double-sided diskette drive; a color display or TV; IBM Enhanced Graphics Adapter, COLORPLUS or equivalent, or REALCOLOR or equivalent; an asynchronous communications adapter; SMARTMODEM 1200 or 1200B or equivalent, WD212-X modem (required for X.25 LAPB support) or equivalent, or a 1200-baud manual-dial modem.

Software

Listen to Learn



Listen to Learn is a talking word processor that helps students develop reading, writing and spelling skills. Educators, therapists and parents can use this product to teach verbal and written concepts to people with a developmental age of at least two years. Listen to Learn can aid in teaching pre-school and primary school age children, students with learning disabilities or communicative disorders and students learning English as a second language.

Teachers can choose from several options in presenting material to students. Passages of text displayed on the computer monitor can be simultaneously spoken. Students can type their own pieces of text, then listen as it is spoken to them. Both methods help reinforce the students' knowledge of the sounds, spellings, words and ideas. Text and related practice exercises can be tailored to each student's needs and preferences.

Listen to Learn comes with a program diskette, a file diskette and a user's manual. This product runs on all members of the IBM Personal Computer family and requires 128KB of memory; one double-sided diskette drive; IBM Enhanced Graphics Display, IBM Color Display, IBM PCjr Color Display, or color TV with RF modulator; the appropriate adapter for your display; an ECHO PC or other compatible speech synthesizer; an IBM Asynchronous Communications Adapter, an IBM Serial/Parallel Adapter, an IBM PCjr Serial Adapter Cable; and DOS 2.00 or higher.

Missing Letters
Missing Letters is an educational game of language and letters for anyone age 8 and older. The user chooses a passage from over fifty reading selections on the diskette and then chooses from a variety of missing letter patterns to create a puzzle as easy or difficult as desired. An editor allows the user to create new passages and customize patterns for the missing letters. Teachers find customized patterns useful in helping with specific reading problems.

Reading selections include poetry, fables, twistograms, palindromes, passages from classic fiction and informational readings. Patterns of missing letters include every other letter, missing vowels, missing consonants, missing words and blank screens.

Missing Letters comes with one diskette and a booklet. This product runs on all members of the IBM Personal Computer family and requires 128KB of memory; one double-sided diskette drive; an IBM Color Display, IBM PCjr Color Display, IBM Personal Computer Enhanced Color Display, IBM Personal Computer Monochrome Display or a TV; the appropriate adapter for your display; DOS 2.00 or higher and a BASIC cartridge if using the IBM PCjr.

I Can Be Anything
I Can Be Anything encourages users age 8 and older to think about what it's like to work in different occupations by creating related pictures. The pictures are made by choosing from hundreds of predesigned shapes that are grouped according to the specific vocations of Artist, City Planner, Astronaut, Architect and Civil Engineer. Users also can create vocations of their own choosing. Suggested follow-up activities are given for learning more about specific occupations.

Exercises lead the user in creating pictures appropriate to each vocation. For example, the City Planner activities involve altering the environment to suit the changing population's needs as a result of the town's growth. The City Planner's picture shapes include different basic town and city layouts, including downtown, airports and buildings such as museums, schools and city hall.

Picture shapes for the Artist include basic components and varying line widths and textures. The Architect's picture shapes include different house parts, including windows and doors, unusual roof types and pillars and the symbols needed to create mock blueprints. Shapes for the Civil Engineer include bridges, roads and airport components. The Astronaut's picture shapes include parts of spaceships and earth-bound launch pads, as well as planets and other objects found in outer space. Shapes can be drawn in different sizes and colors. Finished pictures can be printed with an IBM Graphics Printer or IBM Color Printer.

I Can Be Anything comes with one diskette, a learning activity book and a command card. This product runs on all members of the IBM Personal Computer family and requires 128KB of memory; one double-sided diskette drive; either an IBM Personal Computer Color Display, IBM PCjr Color Display, IBM Personal Computer Enhanced Color Display or a TV; and the appropriate adapter card for your display.

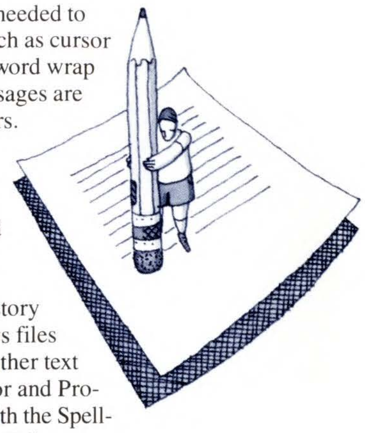
Logo Learner
Logo Learner is a programming language for children and adults who want an introduction to programming. The emphasis on structured programming promotes logical, sequential thinking.

Exercises acquaint students with the Logo language, and sample programs helps students understand how to create their own programs. The Logo Learner editor makes it easy to write and revise programs. By using commands to trace program execution, students can find and correct programming errors.

A text formatter allows the user to edit, indent, insert spaces and include personal comments to help organize programs. Three screen modes — text, graphics and mixed — provide variety in screen presentation. Logo Learner takes advantage of the IBM PCjr's music and graphics capabilities, allowing the user to create colorful graphics and animation from the 16-color palette and three-voice musical sounds.

Logo Learner includes a language diskette, a tutorial and a reference manual. This product runs on all members of the IBM Personal Computer family and requires 128KB of memory; one double-sided diskette drive; an IBM Enhanced Color Display, IBM Color Display, IBM PCjr Color Display, or a TV; the appropriate adapter for your display; and DOS 2.00 or higher. Optional equipment includes an IBM PCjr Compact Printer, IBM Graphics Printer or any compatible printer, IBM PCjr Adapter Cable for Serial Devices, IBM PCjr 128KB Memory Expansion Attachment, IBM PCjr Attachable Joystick, IBM Game Control Adapter, and a compatible external speaker for the PCjr.

Primary Editor
Primary Editor teaches editing concepts to children four years and older, motivating them to practice writing and reading skills. Through graphics depiction and animation, children are introduced to text editing concepts and the keys needed to do simple text editing tasks such as cursor movement, insert and delete, word wrap and page scroll. Friendly messages are given for handling minor errors.



Primary Editor has seven menu options: Write a new story, Read my story, Change my story, Print my story, Read other stories, Start over, and Good-bye. Children's new or revised stories are saved on a story diskette. Primary Editor allows files to be transferred to and from other text editors, such as Personal Editor and Professional Editor. When used with the Spellcheck in Word Proof, Primary Editor provides spelling enrichment.

When a prototype of Primary Editor was used in selected schools for one year, teachers claimed that children found it easy to use. They wrote more and longer stories, revised more and showed a greater desire to write and read. Teachers of Writing to Read students found Primary Editor a particularly appropriate and successful follow-on for their students.

Primary Editor comes with two single-sided diskettes and an adult's manual. This product requires an IBM PCjr Enhanced Model, IBM Personal Computer, IBM Personal Computer XT, IBM Personal Computer AT or IBM Portable Personal Computer with 128KB of memory; an IBM PCjr Color Display, IBM Enhanced Color Display or IBM Color Display; the appropriate adapter for your display; and DOS 1.10 or later. The IBM PCjr Speech Attachment is optional.

PRIVATE TUTOR COURSES

The following group of Private Tutor courses helps students review the basic skills of reading comprehension, language use and mathematics while practicing taking standardized achievement tests. These courses are ideal for students in grades 6 and above and for anyone who wants to improve related skills.

Each of these products comes with two course diskettes and a user's guide. Each is a Private Tutor 2.00 course, requiring Private Tutor 2.00. These products run on all members of the IBM Personal Computer family and require 128KB of memory; one double-sided diskette drive; an IBM Enhanced Color Display, IBM Personal Computer Color Display, IBM PCjr Color Display, IBM Monochrome Display, or a TV; the appropriate adapter for your display; and DOS 2.00 or higher.

Vocabulary Building Skills
Vocabulary Building Skills reviews rules and techniques used in building a better

understanding of words and how they are formed and used. The six lessons cover compound words, words that have a Greek heritage, word roots, prefixes and suffixes.

Word Knowledge Skills
Word Knowledge Skills reviews techniques for understanding the meaning and use of words. Six lessons cover words with multiple meanings; homonyms, synonyms (including how to choose precise synonyms by considering a word's connotative meaning), antonyms, and idioms.

Preparing for Geometry and Algebra
Preparing for Geometry and Algebra reviews the rules and techniques used in introductory algebra and geometry problems. The six lessons offer problems concerning measurement, introductory geometry, graphs, integers, introductory algebra and mixed topics such as money, time, calendar, temperature, Roman numerals and number sequences.

Solving Math Word Problems
Solving Math Word Problems reviews the techniques used to solve math word problems. The five lessons cover word problems that can be solved with one or two computations, word problem analysis, word problems involving percentages, and general applications including money problems, averages, decimals, fractions and distance-rate-time problems.

Math Computation Skills
Math Computation Skills reviews basic mathematical operations. Five lessons cover whole numbers, decimal numbers, addition and subtraction of fractions, multiplication divisions of fractions, and ratios and percentages.

Basic Number Concepts
Basic Number Concepts reviews the usage and nature of numbers. The four lessons cover place value of digits, expanded notation, number lines, rounding and estimating, multiples and factors, even, odd and prime numbers, the greatest common factor, the least common multiple, the least common denominator, simplifying fractions, number sentences including equations and inequalities, number properties including the commutative, associative and distributive properties, and identity elements and inverses.

Reading Comprehension Skills
Reading Comprehension reviews the techniques used to analyze and understand written text. The five lessons cover literal comprehension (information), interpretive comprehension (ideas such as cause and effect relationships), character analysis (feelings, motives, traits, etc.), critical comprehension (author's attitude, viewpoint, and techniques of persuasion), and figurative language (similes, metaphors, hyperbole and personification).

BIOLOGY SERIES

The Biology series of courses is a set of integrated science programs designed by teachers to help teenage students learn the concepts and processes necessary to understand living systems. The varied presentational techniques offer students a logical, investigative approach that emphasizes thinking skills and the scientific process.

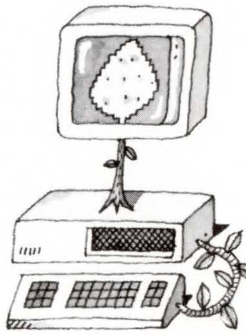
The highly interactive instruction features color graphics and simulations. Each program covers a major conceptual area and is designed to complement the basic biology curriculum used in schools. The courses allow the students to pace themselves through the lessons, offering quizzes and tests in each course.

Each Biology Series course comes in two types of packages: an individual pack consisting of one guide book and one diskette, and a school pack consisting of one guide book and twelve copies of the same diskette. These products run on all members of the IBM Personal Computer family and require 128KB of memory; one double-sided diskette drive; an IBM Personal Computer Color Display, IBM PCjr Color Display, or a TV; the appropriate adapter card for your display; and DOS 2.00 or higher. The IBM Graphics Printer is recommended.

Cell Functions: Growth and Mitosis
The Cell Functions: Growth and Mitosis course identifies the parts of a typical cell, the key structures of the cell and the roles they play in the process of mitosis, and the sequence of events during cell mitosis. The course describes the function and structure of a cell membrane; explains how the surface area and the volume of a cell are determined, the relationship of waste and nutrient transport to cell surface area and volume, how the size of a cell is a function of its volume and surface area, how DNA replicates, and what the significance is of a specific chromosome number for each species; and defines homologous chromosomes.

Light, Plants and Photosynthesis: Energy in Conversion
The Light, Plants and Photosynthesis: Energy in Conversion course describes the process of photosynthesis — how light energy is converted to food energy; identifies the reactants and producers needed during photosynthesis; explains the role of light in photosynthesis, why different colors of light are either reflected or absorbed, and the role of chloroplasts in light conversion; constructs an experiment that proves light is necessary for photosynthesis; and defines controls and variables within the experiment.

Plants: Growth and Specialization
The Plants: Growth and Specialization course describes the undifferentiated meristematic tissue and how it develops into palisade cells; the structure of a stem cross section including wood, bark, pith, cambium, phloem and xylem; the function of xylem, phloem and cortex tissue; and the development of cambium tissue into xylem and phloem. The course defines specialization and differentiation; identifies the zones of differentiation in a root tip and a terminal bud; explains the effects of hormones on plant growth, the relationship of surface area to increased absorption in root hairs, and the role of phototropic responses of plant stems; and interprets a graph depicting inhibition versus plant growth.



Chemicals of Life I: The Structure of Matter
The Chemicals of Life I: The Structure of Matter course identifies the parts of a typical atom, including the proton and neutron (nucleus), and the electron shell; describes the electrical nature of matter and the Bohr model of atoms; calculates the total charge of an atom, its atomic number, and atomic mass; identifies an element's symbol from the periodic table; gives orbital patterns within atomic structures; explains ion formation, oxidation states, and stable octets, and how valence electrons affect the formation of compounds; describes the particulate nature of a molecule; explains that molecules have the chemical properties of compounds they make up; demonstrates the difference between ionic and covalent bonding; and when presented with the structural formula of a compound, determines the number of particular atoms in it.

Chemicals of Life II: Water, Carbohydrates and Lipids
The Chemicals of Life II: Water, Carbohydrates and Lipids course identifies the elements essential to all living organisms, explaining the differences between organic and inorganic compounds; explains the role of water in biosynthesis; illustrates the reactions of ionization involving acids, bases and salts; describes an experiment which illustrates how starch is digested; explains the origin, identity, and function of carbohydrates and lipids in living systems and how hydrolysis and dehydration synthesis are involved in the formation and breakdown of carbohydrates and lipids; gives examples of common carbohydrates and lipids found in living systems; shows the function of ADP and ATP in cellular respiration; gives examples of common carbohydrates and lipids; and explains the difference between saturated and unsaturated fats and what changes the level of cholesterol in the human body, emphasizing its significance in maintaining good health.

Modern Genetics: Chromosomes and Coding
The Modern Genetics: Chromosomes and Coding course describes the chemical makeup of a chromosome; lists the components of nucleotides; explains how DNA is the hereditary control center of the cell and how DNA replicates; describes the function of ribosomes in protein synthesis, showing how the genetic code determines the products that are made in the body; depicts the role of RNA in protein synthesis, explaining the synthesis pathway by which triplet codons determine the sequence and length in polypeptide formation; gives examples of hereditary diseases, showing how they are inherited; defines the different types of mutations and mutagenic agents; and explains how cystic fibrosis and sickle cell anemia affect an individual.

Human Life Processes I: Cellular Physiology
The Human Life Processes I: Cellular Physiology course lists the characteristics of living organisms; defines the life processes of regulation, synthesis, excretion, growth, nutrition, respiration, reproduction and transport; describes the function and structure of the plasma membrane, listing the factors that control the passage of molecules through cell membranes; defines active transport, phagocytosis and pinocytosis; describes an experiment that illustrates oxidation, explaining the difference between oxidation and reduction; describes the role of ATP and mitochondria in cellular respiration; and shows the pathways of anaerobic and aerobic respiration.

Leaf: Structure and Physiology
The Leaf: Structure and Physiology course describes the structure and function of the petiole, stem, veins, midrib and blade; explains the difference between xylem and phloem tissue, how water and food is transported in plants and the function of each layer of the typical leaf; defines the process of photosynthesis, listing the materials needed and the products obtained from photosynthesis; describes the structure and physiology of a chloroplast, the structure of a stomate, and how varying concentrations of glucose and water affect the opening and closing of stomates.

Passive Transport: Diffusion and Osmosis
The Passive Transport: Diffusion and Osmosis course defines diffusion and osmosis, describing how some molecules pass through selectively permeable membranes; explains how altering salt concentration affects osmosis and how equilibrium is achieved on a cellular level; defines such terms as hypotonic, isotonic and hypertonic to explain changes in cell size due to osmosis; and predicts the effect of placing plant and animal cells in solutions of different salt concentrations.

Pathology: Diseases and Defenses
The Pathology: Diseases and Defenses course defines the difference between infectious, genetic and noninfectious diseases, including such noninfectious diseases as heart disease, dietary deficiency, genetic disorders, cancer, stroke and high blood pressure; identifies disease-producing organisms through pictures and descriptions, including bacteria, viruses, protozoa, fungi and worms; explains how disease-producing organisms enter the body and how a virus replicates itself; describes the three lines of natural defenses (immune system, skin and white blood cells) and tells how each works; describes how antibodies form in reference to antigens, explaining what antibiotics are and how they function; explains passive and active immunity; identifies the consequences of chromosomal breakage (deletion, insertion, inversion and translocation); and illustrates and describes the process of fertilization of normal gametes and contrasts it with abnormal gamete formation due to non-disjunction.

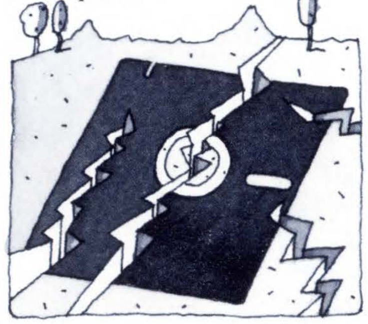
EARTH SCIENCE SERIES

The Earth Science Series is a set of integrated science programs designed by teachers to help teenage students learn the general concepts and processes necessary to understand their environment and what affects it. The highly interactive instruction features color graphics and simulations, offering students a logical, investigative approach that emphasizes reasoning skills as well as natural and scientific processes. Each program covers a major conceptual area and is designed to complement earth science curriculum used in geology, geography and environmental studies. The courses allow the students to pace themselves through the lessons, quizzes and tests given in each course.

The two new Earth Science programs run on all members of the IBM Personal Computer family and require 128KB of memory; one double-sided diskette drive; an IBM Enhanced Graphics Display, IBM Personal Computer Color Display, IBM PCjr Color Display, or a TV; the appropriate adapter card for your display; BASIC cartridge (if using the IBM PCjr); and DOS 2.00 or higher (2.10 for the PCjr).

Volcanoes
Volcanoes includes two graphic tutorials that depict volcanic processes and explain their effect on people and the environment by describing how the distribution of volcanoes is related to plate tectonics; describing the characteristics of composite, shield and cinder cone volcanoes; explaining benefits as well as the direct and indirect hazards of volcanic eruptions; and telling how volcanic activity is forecast.

Earthquakes



Earthquakes includes two graphic tutorials that illustrate how earthquakes happen; explains their effects on people and the environment by showing the relationship of earthquakes to plate tectonics; explains the nature of seismic waves; describes the direct and indirect hazards of earthquakes; and tells how earthquakes may be predicted.

Ask IBM

(Editor's note: Exchange *will publish technical questions and answers from time to time. We are able to respond to only a small number of technical questions, and will publish questions and answers of general interest. If you have a technical question of limited interest, or if you need an immediate answer, we encourage you to ask members of your user group or your Authorized IBM Personal Computer Dealer.)*

Q: DOS Under DOS 2.10 and DOS 3.00, what is the maximum number of files I can have open at one time?

A: The maximum number of files allowed in the entire system is 99 for DOS 2.00/2.10 and 255 for DOS 3.00/3.10. (Use the FILES command to establish your maximum.) The maximum number of files you can open concurrently is 20, including 15 user files and the 5 predefined handles for standard input, output, error, auxiliary, and standard printer.

If you specify FILES= in your CONFIG.SYS file, the size of the resident portion of DOS increases by 39 bytes in DOS 2.00/2.10 and 48 bytes in DOS 3.00/3.10 for each additional file above the default value of 8.

You can open more than 15 concurrent user files by using DOS function calls 4A and 4B to load and execute another process (program). Each time you issue these function calls, you add 15 more concurrent user files. You can continue in this manner until you reach the maximum of either 99 or 255. (For details, refer to function calls 4A and 4B in the DOS manual.)

Q: I have recently expanded my PC to 640KB of memory. APL now hangs the machine and must be hard-booted. How can I get APL to work on a 640KB system?

A: Temporarily re-configure your system for 512KB by changing the system switch setting. Then enter the following patch to the APL program diskette. (User inputs are in quotes, and <ENTER> denotes pressing the Enter key.)

```
B> "APL AP210 <ENTER> "
CLEAR WS
    ")IN FILE <ENTER> "
    "PATCH 'APL.EXE' <ENTER> "
    GIVE ADDRESS: "37A <ENTER> "
    IS 7D
    GIVE NEW VALUE OR EMPTY LINE
    TO
    CANCEL PATCH
    : "73 <ENTER> "
    GIVE ADDRESS: "<ENTER> "
    ")OFF <ENTER> "
```

You should now be back in DOS and ready to run APL. Reset your system switch settings to their original values.

Q: I have a serial printer connected to my PC. When I utilize the TopView DOS Services function to print, the output is sometimes erratic.

A: Unlike the DOS PRINT command, the TopView DOS Services PRINT function does not expand tabs. Also, a serial printer does not expand tabs. Therefore, when you use a serial printer with TopView DOS Services PRINT, the result may be unpredictable.

Calling All Computing Theory Articles!

Exchange carries a number of technical articles about specific IBM Personal Computer hardware and software. However, the subject of computing also involves general concepts that apply to many different computers and kinds of software.

In Exchange, we classify conceptual material under the heading *Computing Theory.* When Bernard Penney was the editor of the diskette version of Exchange, he noted that PC user group newsletters sometimes contained well-written material of a conceptual or theoretical nature. As a result, Bernard implemented a Computing Theory section that turned out to be one of the more interesting parts of the diskette. We are continuing with the Computing Theory section in the new Exchange.

We welcome user-written contributions to all sections of Exchange, but we especially need articles for the Computing Theory section. It takes a special breed of person to think about microcomputing theory and (more important) to transpose thoughts into clear, understandable writing. Such articles appear less often than other kinds, which is why Exchange has no Computing Theory articles this month. However, I feel confident that user groups have talented members who can theorize and write.

Therefore, I call upon readers of Exchange to contribute microcomputing theory articles to their user group newsletters and send the newsletters to us. We read all newsletters we receive; we will see your theory article, and if it is worthwhile and interesting, we will (with your permission) publish it in Exchange.

Michael Engelberg
Editor

Copyrights, Trademarks and Service Marks

"The ECD is easy to read and the colors are good and clear. (page 2)

"Programs that use copy protection schemes should use the ROM BIOS diskette calls to read and verify the diskette and should not be timer dependent. (page 6)

"Use IBM-supported high level languages whenever possible. (page 7)

"The DOS Technical Reference Manual has been reorganized. Many chapters have been expanded and others are new. (page 18)

"You'll find that BATch files can simplify and enhance not only your setup procedures, but almost any repetitive task. (page 20)

"Be sure to take your sales receipts and invoices to the international airport and file U.S. Customs documents before leaving on a trip. (page 34)