

# PERSONAL SYSTEMS

## Hardware

- 1** Low-Cost Innovation: The PS/2 Model 30
- 5** The IBM 8507 Monochrome Display
- 6** Inside the IBM Proprinter III

## Software

- 9** OS/2 Capacity Planning
- 20** Optimizing Your Use of the DOS 4.00 Shell
- 28** DOS Shell Program Startup Commands
- 32** Discover/Education: Learning with a Difference

## Random Data

- 39** Performance Improvement: A Case Study
- 42** 3.5-Inch Diskette Formats
- 43** DASD Details
- 50** Making a Mirror Image of a Fixed Disk
- 57** New Products



*IBM Personal Systems Technical Journal* is published by the U.S. Marketing and Services Group, International Business Machines Corporation, Boca Raton, Florida, U.S.A.

Editor	Mike Engelberg
Production Coordinator	Wayne Hammond
Illustrator	Jeff Jamison
Automation Consultant	Andrew Frankford
Manager	Gene Barlow

To correspond with the *IBM Personal Systems Technical Journal*, please write to the Editor at IBM Corporation (4409), P. O. Box 1328, Boca Raton FL 33429-1328.

To subscribe to this publication, use an IBM System Library Subscription Service (SLSS) form, available at IBM branches, and specify form number GBOF-1229.

Permission to republish information from this publication is granted to publications that are produced for non-commercial use and do not charge a fee. When republishing, please include the names and companies of authors, and please add the words "Reprinted by permission of the *IBM Personal Systems Technical Journal*."

Titles and abstracts, but no other portions, of information in this publication may be copied and distributed by computer-based and other

information-service systems. Permission to republish information from this publication in any other publication or computer-based information system must be obtained from the Editor.

IBM believes the statements contained herein are accurate as of the date of publication of this document. However, IBM hereby disclaims all warranties as to materials and workmanship, either expressed or implied, including without limitation any implied warranty of merchantability or fitness for a particular purpose. In no event will IBM be liable to you for any damages, including any lost profits, lost savings or other incidental or consequential damage arising out of the use or inability to use any information provided through this service even if IBM has been advised of the possibility of such damages, or for any claim by any other party.

Some states do not allow the limitation or exclusion of liability for incidental or consequential damages so the above limitation or exclusion may not apply to you.

This publication could contain technical inaccuracies or typographical errors. Also, illustrations contained herein may show prototype equipment. Your system configuration may differ slightly.

IBM has tested the programs contained in this publication. However, IBM does not guarantee that the programs contain no errors.

This information is not intended to be a statement of direction or an assertion of future action. IBM expressly reserves the right to change or withdraw current products that may or may not have the same characteristics or codes listed in this publication. Should IBM modify its products in a way that may affect the information contained in this publication, IBM assumes no obligation whatever to inform any user of the modifications.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming or services in your country.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever.

All specifications are subject to change without notice.

# Low-Cost Innovation: The IBM PS/2 Model 30

Steven Howell  
IBM Corporation  
Boca Raton, Florida

*Editor's note: This article is reprinted from Innovations, an IBM Boca Raton site publication.*

The primary driving force behind the design of the IBM Personal System/2 Model 30 was cost. Conventional methods of mechanical packaging used in the past had not yielded the costs that were necessary to be competitive in the low-end market. Therefore the mechanical engineering design team for the PS/2 Model 30 was challenged to be innovative in a way that would reduce the cost of mechanical packaging.

A task force was established to identify previous problems, define the new criteria, and establish a design concept that encompassed as many good engineering practices as possible. Constituting the task force were experts from a number of organizations, including industrial design, manufacturing engineering, planning, and electromagnetic compatibility (EMC) test engineering.

This article discusses some of the innovative ways that the design objective was met.

## First, The Frame

Typically, one of the most expensive elements of the mechanical structure is the frame. In the past, frames had many welds for strength

and function, adjustable springs for EMC grounding, and masking and painting problems.

The first decision was that the frame would be non-decorative, and not visible in the assembled machine. This permitted the use of less expensive forming techniques, and eliminated the need to paint and mask.

*The primary driving force behind the design of the IBM Personal System/2 Model 30 was cost.*

The second innovation was new in the design of personal computers: to use extruded material wherever possible to eliminate strengthening welds. (See Figure 1, Items A and C.)

At this point, many more uses for this technology were realized. Item B is a drawn-up pad with a pierce and extruded hole used for mounting the planar board. This eliminated the need for nylon stand-offs, which had been used in the past.

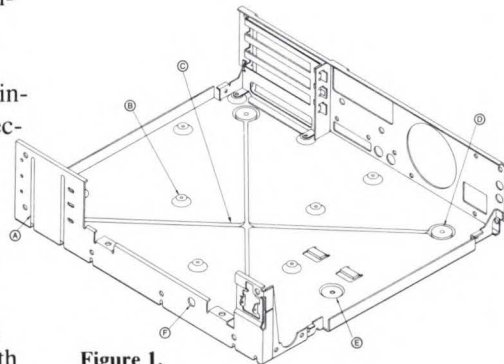


Figure 1.

Item D is a ring configuration used for locating the feet, which eliminated the need for templates, and greatly reduced assembly time.

Item E is extruded down to the same depth as the feet. This feature was provided to enable schools to bolt the system to the table to prevent damage or theft.

Item F is a clearance hole that enables an alignment pin from the front bezel to engage with the DASD support cradle, further ensuring good alignment between the drive units and the front bezel.

The frame is constructed out of 1.2-millimeter sheet metal with a nickel 20 finish – a long-proven performer in EMC grounding.

## Surface Contact

Another of the inherent problems of the past was maintaining good surface contact between the option card brackets and the shadow box portion of the frame. In the past this was accomplished using expensive springs that had to be set in a fixture or adjusted at assembly. Both of these methods had a common flaw – they pushed at the bottom of the bracket where it was not supported, causing it to bow away from the contact surface, which resulted in EMC gaps.

The Model 30 design has no springs and no adjustments. All the card brackets have a five-degree bend approximately 10 mm from the bottom of the bracket. This feature was originally designed to assist in the card insertion as a lead-in.

The design illustrated in Figure 2 uses this bend in the card bracket as its spring. As the bracket is inserted, it is supported for its entire

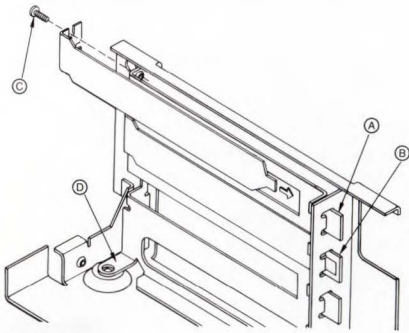


Figure 2.

length by surface item A. As the bracket reaches its final position, the curl bend of the shadow box is at the peak of the five-degree bend, and because the bracket is fully supported by Item B, it is forced against the contact surface of the shadow box. The bracket is further secured by the traditional screw, item C. Item D shows a direct ground connection from the shadow box to the planar board mounting / grounding pad.

### Option Card Mount

The option cards were mounted horizontally, for many reasons. Horizontal mounting allows op-

imum use of planar space, and because printer and video functions were integrated on the planar, connector room was needed along the back edge of the planar board.

### *The option cards were mounted horizontally, for many reasons.*

In order to connect the option cards to the planar board, a bus card would be necessary. Despite the cost of a bus card, many advantages were soon realized. The horizontal orientation of the option cards allowed a lower-profile box, and it improved thermal and acoustic performance. Also, the bus card offered a convenient place to mount the battery without having to handle a small cable during assembly.

The main problem that arose with the bus card configuration was its

need to be mechanically supported during insertion and extraction of the option cards. Another innovative solution was found, as shown in Figure 3. A plastic retainer support bridge (item B) goes into the side of the power supply through two specified slots. It pivots downward and snaps over the bus card. It is retained by the uppermost connector. Because the forces of insertion and extraction are applied to the compression and tensile properties of the plastic material, the bus card is held rigid. The benefit of this is that the stresses are taken off the bus card and, more importantly, the planar board and its surface-mounted components.

To remove the bus card or to gain access to the user area below it on the planar board, the user simply squeezes the two tabs (item A) at the top and lifts on the handle between them, pivoting the support bridge up and out of the way.

Another problem in horizontal mounting appeared at the other end of the option cards. In the past, single card guides were used. They had a tendency to rock because they were supported at only two points. Also, the time involved in inserting each individual guide was costly. All three slots were incorporated into a single unit. This provided much more rigidity and required

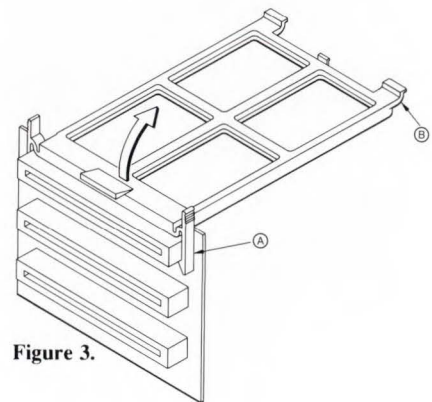
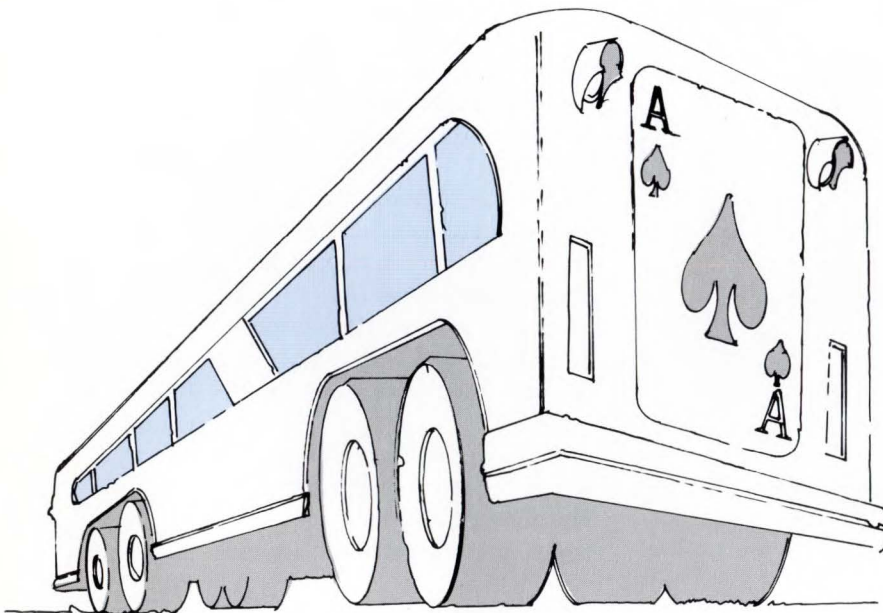


Figure 3.

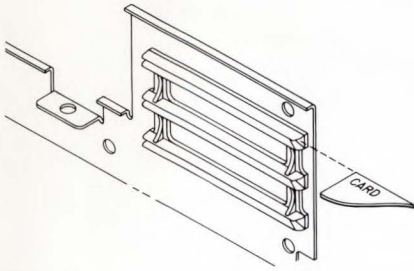


Figure 4.

less assembly time. Figure 4 illustrates the new card guide.

Throughout the entire design process, ease of assembly was sought rather than automation. Automation requires stricter tolerancing of mating parts, which increases the cost of the parts and requires a large capital investment on the manufacturing line. It was decided that manual assembly, combined with parts that were easy to handle, would result in a lower assembled cost.

### On / Off Switch

A high-priority item for all the newly developed products was moving the power supply on / off switch to the front of the unit. This was done to provide easier user access, and rack or furniture mounting of the system unit. The size objectives precluded running a power supply from front to rear without increasing the cost of the power supply. Also, running a tethered electrical cable to the front of the

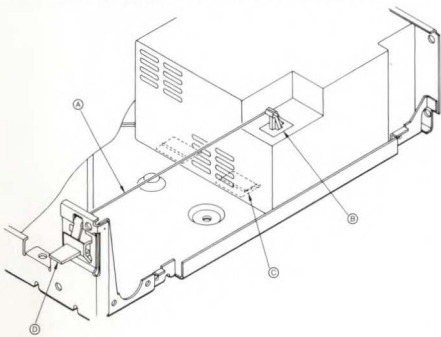


Figure 5.

machine arouses EMC and safety concerns.

By using a remote actuated toggle, shown in Figure 5, good appearance alignment with the front bezel could be maintained, and EMC and safety concerns minimized. The power supply was mounted to the frame by means of two tabs on the bottom and three screws from the rear. A plastic toggle (item D) was mounted to the front of the frame in a minimum amount of space. The power supply (item C) has a non-decorative switch (item B) mounted on the front corner, which is activated by a pivoting linkage (item A). Tolerances front-to-rear are absorbed by the

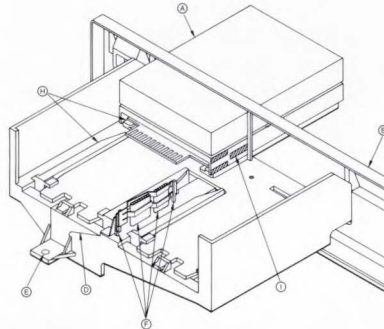


Figure 6.

"over-travel" of the functional switch, and by the flexible characteristics of the stainless steel wire linkage. The linkage snaps into place to facilitate the assembly and replacement. The decorative toggle is pivoting on an aluminum pin, held in place by the frame. In all, this is a simple design that works well.

### Cradle and Bezel Design

Figure 6 illustrates the DASD support cradle and front bezel. The drive unit (item A) was mounted to a slide that engages with tracks (item H) provided by the cradle. As the drive unit reaches its final position, the slide latches to the cradle. Presently, the drive unit is grounded

by means of plated plastic slides and support cradle.

Items D and F illustrate a planned means of eliminating the expensive plating of plastic, by using a leaf spring arrangement (item J), which would contact the drive unit at three points (item I), and ground to the single cradle mounting screw (item E). Because the spring floats, it provides equal pressure to the two drive units and requires no adjustment.

Figure 7 shows the front bezel and support cradle design features in more detail. Although they are two separate parts, they form an integral structure. The support cradle is mounted to the frame, aligned by two pins (item C). The front bezel is pushed onto the frame from the front, and aligned with the cradle via the alignment pin (item F). The bezel snaps into the frame and locks integrally with the cradle by means of two large snaps (item A) to the respective catches (item B).

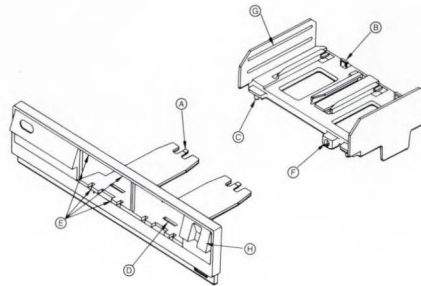


Figure 7.

Item G indicates two tracks in the side wall of the cradle that an adjustable card support slides in. This is used when the system unit will be shipped with special configuration option cards that are not full-length. Therefore they require additional support at their ends. Because the length of the cards varies from op-

tion to option, the card support must adjust for the different lengths.

The drive unit slides catch on the slots (item D), requiring no screws to install the drives. The decorative drive bezels snap to the front bezel

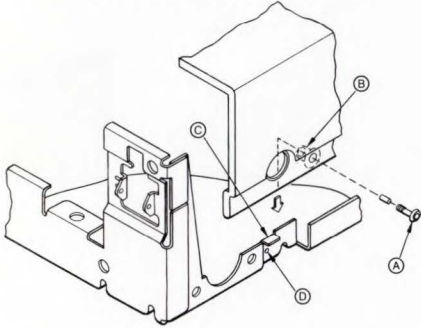


Figure 8.

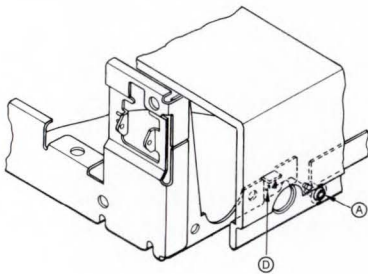


Figure 9.

at points marked E. Because the bezel and cradle are locked together, it is possible to maintain tight alignment, which also aids the appearance. Item H is a molded-in switch bezel for the power-on toggle. Icons are molded in and, as with the drives, the alignment is held tightly.

### Covers

For appearance purposes, the decision was made to use a plastic top cover. This also allowed the incorporation of a great many desirable features into the cover design. Figure 8 shows the right front corner as an example of the four cover screws. A captive spring-loaded screw (item A) is inserted through the top cover (item B). The top cover drops down on the chassis 10 mm toward the rear. The cover bosses (item B) locate under the chassis tabs (item C) as the cover is slid 10 mm forward to its final position (see Figure 9).

Large flat plated areas on the frame contact the cover along both sides, and positioning of the screws at the four corners of the cover ensures a

good EMC design. The amount of space required to remove the cover is only slightly larger than its footprint. The tabs and molded channels make the machine more temper-resistant.

Finally, because the frame is non-decorative, the rear of the machine needed a fascia, which was made by molding a graphite gray plastic bezel to snap in place on the rear panel. Icons and logo were molded in, as well as knockouts for option card cable exits. The cost was minimal, considering the value added to the appearance of the bezel. Figure 10 shows the system unit in an exploded illustration.

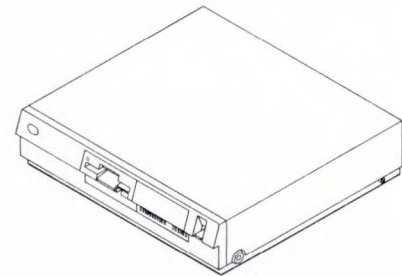


Figure 11.

Another goal achieved in this design is serviceability, the ease in which a field-replaceable unit can be removed and replaced. The planar board, for instance, can be unscrewed and slid out the left side. The power supply can be unscrewed and the cables disconnected and removed from the top without removing the uninvolved components. The same is true for the drive units.

Figure 11 shows the system unit in its final assembled configuration.

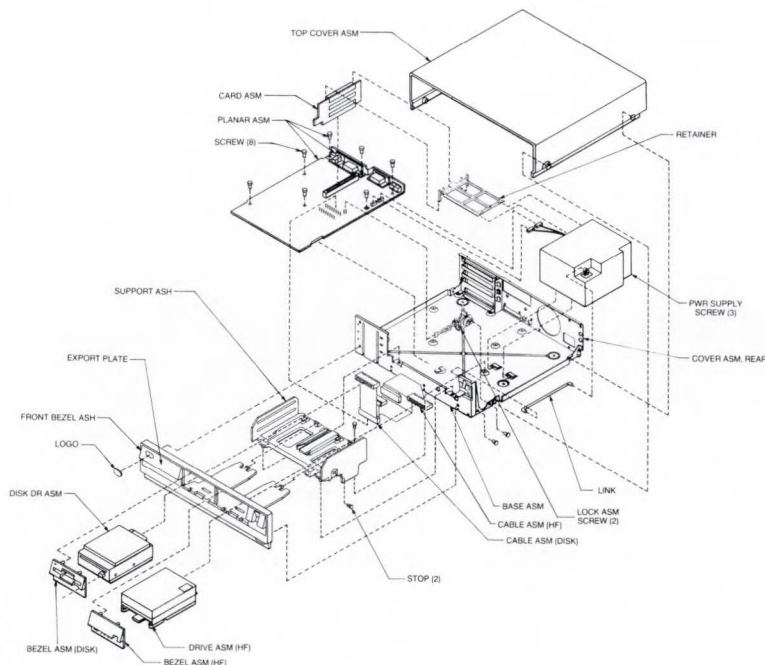


Figure 10.

# The IBM 8507 Monochrome Display

Kevin Maier  
IBM Corporation  
Boca Raton, Florida

If your application requires advanced function, high resolution, and high performance, but does not require color, the new IBM 8507 Monochrome Display may fill the bill.

The 8507 Monochrome Display is designed for use on Personal System/2 computers that are equipped with Video Graphics Array (VGA) display technology. The display supports all VGA modes the same way as the 8503 Monochrome Display does, but the 8507 has a larger screen.

The 8507 is a monochrome version of the IBM 8514 Color Display, with a larger viewing area and a universal power supply.

The 8507 Monochrome Display can be attached to the 8514/A Display Adapter in PS/2 systems. When attached to the 8514/A adapter, the 8507 displays the higher resolution that the adapter generates.

When used with the 8514/A Display Adapter and the optional graphics memory expansion option for the 8514/A adapter, the 8507 can display advanced graphics, text, and image in scientific, engineering, and host-interactive environments. Its 19-inch screen and its ability to display 146 columns by 51 rows offers advantages for applications such as spreadsheets and desktop publishing.

The 8507 display can also be attached to a PS/2 Display Adapter installed in an IBM Personal Computer, Personal Computer XT, or Personal Computer AT, and to the MCGA port on a PS/2 Model 30.

## Specifications

The 8507 is a 19-inch, analog, multi-mode, white phosphor, raster display capable of displaying up to 1024 x 768 pixels at 73 pixels / inch in a viewable area of 14 inches x 10.5 inches. The display can show from 16 to 64 shades of gray, depending on the video mode being used.

*The display can show from 16 to 64 shades of gray.*

The 8507 display has a universal power supply similar to the one contained in PS/2 system units, and can be used with different power sources without any switching (other than On/Off). The 8507 has an anti-reflective screen for reducing glare, and a tilt/swivel pedestal for operator comfort and convenience.

The 8507 also departs from the standard PS/2 display layout in that all controls are located below the screen on the front of the unit.

The following are the specifications for the 8507 display:

- Vertical addressability of 350, 400, 480 or 768 lines.
- Video bandwidth of 45 MHz.

- Self-test with a white screen test pattern.
- 75-ohm, direct-drive analog video input 0.0 Vdc - 0.7 Vdc.
- 19-inch cathode ray tube.
- Etched, anti-reflective faceplate.
- Medium persistence phosphor.
- Horizontal deflection rate of 31.5 kHz or 35.5 kHz in 1024 x 768 mode.
- Horizontal blanking time of 5.7 microseconds (usec), or 5.35 usec in 1024 x 768 mode.
- Vertical deflection rate of 70 Hz (350 and 400 line mode), 60 Hz (480 line mode), or 43.5 Hz interlaced (1024 x 768 mode).
- Vertical blanking time of 2.7 milliseconds (msec) in mode 1, 1.1 msec in mode 2, 0.92 msec in mode 3, and 0.69 msec in mode 4. Modes refer to the number of scan lines in the active display area. The four modes are:
  - Mode 1 = 768 scan lines (interlaced)
  - Mode 2 = 350 scan lines (non-interlaced)
  - Mode 3 = 400 scan lines (non-interlaced)
  - Mode 4 = 480 scan lines (non-interlaced)
- 6-foot signal cable with a miniature 15-pin, D-shell connector.
- 6-foot, detachable power cable.
- Size:
  - Width: 18.75 inches
  - Depth: 17 inches
  - Height: 17.75 inches
- Weight: 52 pounds

# Inside the IBM Proprinter III

Gary Brown and Jess Blackburn  
IBM Corporation  
Charlotte, North Carolina

Two new models of the IBM Proprinter have expanded IBM's family of 9-wire, dot-matrix printers. The IBM Proprinter III and the IBM Proprinter III XL are ideal for customers who require near-letter-quality (NLQ) printing, all points addressable (APA) graphics, and fast throughput.

While the appearance of the new models has changed very little, the engineering and programming effort needed to make the enhancements was significant. For example, about 80 percent of the parts in the Proprinter II were changed to arrive at the Proprinter III.

The 4201 Proprinter III is an enhanced version of the Proprinter II. The Proprinter III has increased print speed, a new italics font, blacker ink in the ribbon, and Propark – a one-touch paper parking function.

A major improvement in the new models is speed. The Proprinter III's burst speed is up to 270 characters per second (cps) in 10 characters per inch (cpi) draft mode. This is 35 percent faster than the corresponding 10-cpi draft mode speed of the Proprinter II and II XL. The new printers also print at up to 320 cps in Fastfont, a special 12-cpi spacing, and at up to 65 cps in near-letter-quality (NLQ) mode. The Proprinter III delivers 50 percent im-

*About 80 percent of the parts ... were changed.*

provement in draft and NLQ throughput. All-points-addressable graphics printing, depending on the image and the density, has increased by up to 400 percent!

The Proprinter III has an optional single-drawer sheet feed. Other options include a larger print buffer and an enhanced serial interface.

The 4202 Proprinter III XL is identical to the Proprinter III except that it has a wide carriage. The Proprinter III XL is an enhanced version of the Proprinter II XL.

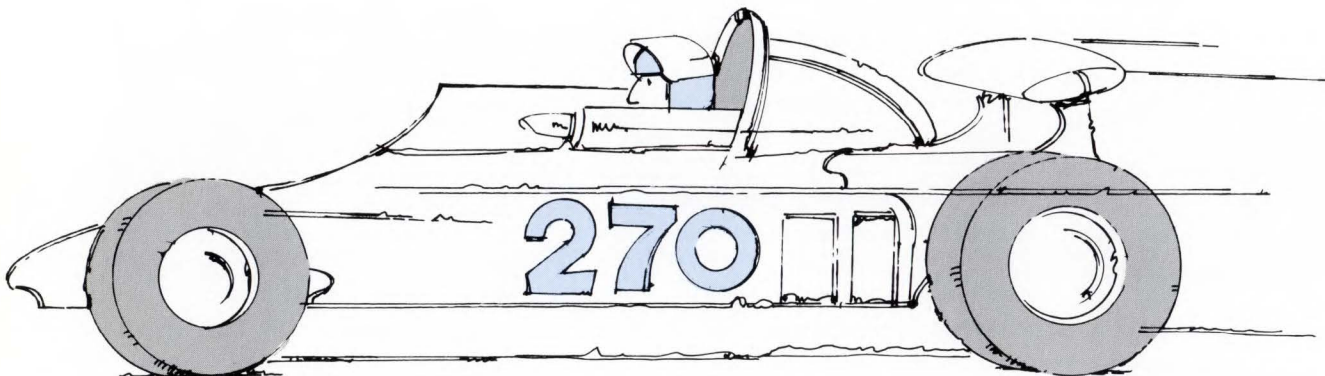
## Ease of Use

Designed and manufactured by IBM, the Proprinter family of personal printers offers quality, versatility, and ease of use.

Tearing off the printed page is one of the most common tasks that anyone using a printer performs. With the new Proprinter III and Proprinter III XL, tearing off a printed page is quick and efficient, using the paper-tear-assist feature. It's as simple as pushing down on the paper cover and tearing off the page just printed – without wasting a page.

## Paper Parking with Propark

The Propark paper-parking function allows continuous forms to remain in a printer without advancing, while cut sheets are inserted. Conventional designs of other printers require the user to manually move covers, levers, bails, and clutches.





With Propark, users can avoid wasting continuous-form paper while using the front-loading feature to print on cut-sheet paper. Changing paper from computer forms to cut sheets on the new Proprinters is as quick and easy as pushing a few buttons. To switch from continuous forms to cut sheets:

(1) Press the Propark button. When this is done, the mechanics within the printer behave like the tumblers in a combination lock. The continuous-forms paper moves backward, then forward, then backward again, and a clutch disengages the paper-advance tractors.

(2) Insert a cut sheet in the front and press the Feed key. Friction rollers move the individual sheet of paper through the printer without moving the continuous-forms paper.

(3) Press the Online button. The cut sheet is now ready for printing.

To switch back to continuous forms:

(4) Press the Propark button. The mechanical "combination lock" process reverses, the continuous-forms paper returns to the starting position, and the paper-advance tractors are re-engaged.

(5) Press the Online button. The continuous forms are once again ready for printing.

### Throughput

When measured by the PC Labs Printer Benchmark Test published in *PC Magazine*, the IBM Proprinter III had approximately 50 percent faster throughput than a leading 250-cps printer when printing at 10 cpi in draft mode.

The actual throughput speed as measured by the Printer Benchmark

Test is 173 cps for the IBM Proprinter III. The IBM Proprinter III and Proprinter III XL are the throughput leaders in their price class.

### *Proprinter III models have a new microcode architecture.*

To do your own throughput test, simply press the Print Screen (PrtSc) key and measure the elapsed time on two different printers.

### Engineering Improvements

Proprinter engineers have minimized the length of time it takes to move the paper and to change the print head direction, for each line. In addition, NLQ printing and graphics printing now take place while the print head moves in both directions.

The Proprinter III models have a new microcode architecture that results in significant improvements in the Proprinter's APA graphics processing.

Another factor in boosting the Proprinter III's speed is a redesigned 9-wire print head. A heat sink was added to dissipate some of the additional heat that results from the faster movement of the print head. Moving the print head faster and, consequently, firing the print head wires faster required a number of related changes, such as a harder platen – the flat metal plate behind the paper that the print wires strike against. Also, the distance between the print head and platen was decreased, and the print unit was stiffened to handle the demands of the high-speed print

head. A new low-cost, high-torque motor was added to the design to improve throughput. The step motor allows for fast stops and starts as the print head zips at up to 270 cps back and forth across the paper.

A unique method detects whether the print head is getting too hot. The Proprinters I and II use an algorithm that calculates (through microcode) the heat of the print head by the number of times that the wires are fired in a particular span of time. For the Proprinter III, a circuit monitors the resistance in an electrical coil in the print head. When the resistance changes enough to indicate that the print head is overheating, it automatically slows the overall throughput for a moment to cool. This approach frees up the microprocessor from having to track the temperature through an algorithm. This makes room for more microcode to further improve the function of the printer.

Although the speed of the Proprinter III is faster, the sound it makes is no louder and, at times, is even softer, than the sound made by the preceding models. Acoustic improvements include isolation of the print mechanism from the external covers and some adjustments to the paper entry and exit paths.

### Space-Age Polymers

All Proprinters make use of some of the most modern space-age polymers:

- Lexan™, used in automotive bumpers.
- Teflon™-Filled Polycarbonate, used in airline seats and security camera housings.

- Acrylic, used in windows and lenses for cameras, telescopes, and eyeglasses.
- Nylon, used in all precision gear applications.
- Glass-Filled Nylon, used in fighter aircraft, missile components, rifles, and radiators.
- ABS Plastic (a copolymer of acrylonitrile butadiene and styrene), used in camera housings and automotive interiors.
- PVC Plastic (polyvinyl chloride), used in pipe, conduit, and automotive interior components.

Lexan is a trademark of General Electric Co.; Teflon is a trademark of E.I. DuPont de Nemours, Inc.

#### **Additional New Features**

Other new features of the Proprinter III and III XL are:

- A 20 cpi print mode prints more columns on a single page.
- An italics NLQ font has been added.
- DIP switches have been eliminated.
- The operator panel has been improved by the addition of pitch lights.
- The enlarged standard print buffer is 7 KB; an enlarged optional print buffer is 32 KB.
- The Proprinter III models are capable of using the same single-bin sheet feeder options that are offered for the 24-wire Proprinters.
- The enhanced optional serial interface now includes both RS-232 and RS-422. RS-422 allows the printer to be located as far as 4000 feet away from the processor.

## OS/2 Capacity Planning

*Steve Berkowitz, Ken Bushby, Betty Johnson, Dave Lacasse, Dave Ogee, and Steve Vaughan IBM Corporation Austin, Texas; Boca Raton, Florida; and White Plains, New York*

The purpose of this article is to clarify the memory and fixed disk storage capacity requirements of OS/2 Standard Edition (SE) and Extended Edition (EE).

For successful capacity planning, it is necessary to anticipate the memory and fixed disk requirements that will be generated by a complete set of system and application components. To assist your efforts, seven typical end-user scenarios are supplied below. These scenarios have been designed to reflect typical OS/2 SE and EE work environments, and they are intended to be used as models for your own analysis.

Beneath each of the scenario charts, a recommended PS/2 system configuration is described. These recommendations are intended to be cost-effective configurations, and are not intended to imply that the scenario shown is restricted to the systems recommended. All scenarios should run successfully on the recommended system as well as on any larger PS/2 configuration that you should choose. Running the scenarios on systems that are appreciably smaller than the recommended configurations could result in performance degradation. Acceptable performance of some user applications may require the larger configurations.

The capacity numbers used for OS/2 system components, which include the base operating system, communications components of the Extended Edition Communications Manager, and the Extended Edition Database Manager, are drawn from the worksheets in Figures 8 and 9. Memory and fixed disk requirements for user applications are based upon the applications available today combined with an estimate of the amount of fixed disk space these applications required in a DOS system.

*For successful capacity planning, it is necessary to anticipate memory and fixed disk requirements.*

Conversion of current DOS programs into OS/2 deserves careful planning. For example, a DOS program that is structured with overlays can be restructured as a single large load module in OS/2. This could result in a larger memory requirement. Depending on the program's execution-time characteristics, this design choice could result in either improvement in, or degradation of performance of, the application. The design choice could also affect other applications that are running concurrently.

Capacity estimates for spreadsheet and word processing applications in the scenarios are based on an informal survey and reflect a minimal requirement for data files. Relatively small requirements for data files are stated in the expectation that many users will store document and spreadsheet files on less expensive

offline media, such as diskettes or file servers.

The most variable element in your capacity planning efforts is likely to be the amount of memory or fixed disk storage required by your unique application set. Your current application data file requirements should be used as the basis for your future planning work.

The design of applications, and the degree to which they use the enhanced functions of OS/2, can result in an increase in memory requirements. The scenarios in this article assume that applications use a basic subset of available function. An application that fully exploits the windows and graphics function of the Presentation Manager could, as an example, generate memory requirements of up to a megabyte more than the base system estimates.

### Interpretation of Memory Requirements

The memory requirement numbers in the "End-User Scenarios" and "Capacity Planning Worksheet" sections in this article do not reflect the total size of all code and data areas of the applications described. To help you understand the rationale behind the decision to present memory requirements in this way, the following background information about the way that OS/2 manages real memory is offered.

OS/2 provides a valuable feature that allows more program code and data to be concurrently active than the available physical memory can contain. This feature, called virtual storage, has important implications for the capacity planner.

Through the use of advanced memory management techniques,

OS/2 attempts to keep only those portions of a program that are actually active in real memory. Inactive portions of a program are placed in a special file, called a Swap Data Set, where they remain until they are needed in real memory. This method allows OS/2 to maintain a greater number of applications in an active state than it would be possible to fit into real memory together if a non-virtual storage system were being used.

Because of this method, only a relatively small part of a typical application or system function tends to remain in real memory. This active part of a program is called its Working Set – the set of program segments actively performing work.

Experience has shown that in virtual storage systems, a significant portion of a program's code and data areas tends to migrate to the Swap Data Set on fixed disk, thereby relieving the demands on real memory.

These facts have been applied in the "End User Scenarios" and "Capacity Planning Worksheet" sections that follow. Notice that in the scenarios the Segment Swap Data Set is shown to be either 2.5 MB or 5.0 MB in size. Because the default start-up size of this data set is 640 KB, it is clear that the end-user scenarios shown are taking advantage of the economies possible with virtual storage. The size of the Swap Data Set indicates that OS/2 is storing inactive portions of programs on disk, and is therefore allowing a larger collective workload to run concurrently than would be possible in a non-virtual storage system.

Due to the memory management techniques of OS/2, the numbers

shown under "Memory" in the scenarios reflect the average working set size of the relevant component, and not the total size of the program. This is a realistic reflection of the way that OS/2 systems work, and it has been chosen here for that reason.

The recommended system configurations that follow each scenario are similarly designed to take advantage of OS/2 virtual storage. Using the recommended memory sizes for the systems will result in an acceptable degree of virtual storage activity for the workloads shown. In all cases, the addition of more memory would result in slightly better performance, as well as the capability of adding more work to the systems.

### *Conversion of current DOS programs into OS/2 deserves careful planning.*

See the sections "Virtual Storage and Segment Swapping" and "Working Sets" later in this article for more information.

### **End-User Scenarios**

Below are several scenarios representing different uses of OS/2 components.

*The recommended configurations were developed using the methodology described previously in the section "Interpretation of Memory Requirements." This methodology assumes an acceptable degree of virtual storage segment swapping during system operation.*

Notes that apply to the scenario charts are:

N/A: Not applicable.

Note 1: Memory size shown is for a single file transfer session.

Note 2: The OS/2 Extended Edition Version 1.1 fixed disk estimate assumes that Configuration Services and API Data Structures were not installed by the user. This implies that the Configuration Files have been supplied to this user from an external source, and that this user does not require the API Data Structures to perform communications application development.

Note 3: The DOS Compatibility application fixed disk requirement is an estimate of the space required for DOS program-resident and associated data files to support the DOS application. It does not represent space required for OS/2 to emulate the DOS environment.

Note 4: This scenario shows one installed database. Each additional database would require 0.5 MB of fixed disk space for system support data, plus the space required for user data.

Note 5: The Advanced Program-to-Program Communications (APPC) fixed disk requirement is included in the fixed disk requirement for 3270 communications.

Note 6: The OS/2 Extended Edition Version 1.1 memory estimate shown for the Communications Manager includes the code required to support the LAN Requestor function.

*Customers assume responsibility for determining and installing the proper memory and fixed disk capacities in their machines; for the selection of the program(s) to achieve intended results; and for the installation, use, and results obtained from the machines and*

program(s). IBM does not warrant or otherwise guarantee any performance or usability results implied by this data. Customers should ensure that adequate hardware configuration flexibility exists to accommodate changes, future enhancements, new customer applications, and increased user data requirements.

#### **Standalone System, No DOS**

**Compatibility:** Figure 1 gives memory and fixed disk requirements for a standalone system on which the DOS compatibility feature of OS/2 is not used.

*Your current application data file requirements should be used as the base.*

Figure 1 illustrates an OS/2 multi-programming system that would support effective concurrent processing of applications. For example, the user could be editing a document, while in the background a spreadsheet is being recalculated and a document prepared earlier is being printed. In this way, the user may experience higher productivity than it was possible to achieve in previous single-task operating environments.

For this workload, a minimum system would be a PS/2 Model 50 with 3 to 4 MB of memory and 20 MB of fixed disk storage.

#### **Communications Manager and No DOS Compatibility:**

Figure 2 shows a system that communicates with other systems, and does not

use the DOS compatibility feature of OS/2.

Figure 2 represents an approximation of a combination of jobs that will be found in many office environments. Here the user can simultaneously perform work on a word processor and a spreadsheet and, at the same time, access the host system.

Productivity improvements can be achieved by making effective use of concurrent activities. For example, the user could be working interactively on the spreadsheet, while in the background the word processor could be paginating a long document, and at the same time the host communications sessions could be downloading a file.

A PS/2 Model 50 Z-031 with 4 to 5 MB of memory and 30 MB of fixed disk would be an appropriate minimum system for this scenario.

#### **Communications Manager Plus the DOS Compatibility Feature:**

Figure 3 is similar to Figure 2 except for the addition of an important new element – the use of the DOS compatibility feature.

Most users can be expected to have an inventory of DOS applications that will be carried over into the OS/2 system. The DOS compatibility feature of OS/2 allows the execution of many of these DOS programs concurrent with the multi-programming mix, which here includes word processing, spreadsheet, and a host communications session. Refer to "DOS Compatibility Mode Structure" for more information.

The DOS compatibility feature permits a gradual transition to OS/2 without having to wait for the con-

version of existing DOS applications into OS/2's native method of operation.

*Virtual storage has important implications for the capacity planner.*

This workload mix would run comfortably on a PS/2 Model 50 Z-031 or 50 Z-061 with 5 MB of memory and 30 MB of fixed disk. However, it should be recognized that this user, who is running four applications concurrently, may be on a fast growth track. If this is the case in your installation, it is likely that a PS/2 Model 60-041 with 5 MB of memory and 44 MB of disk, or a PS/2 Model 60-071 with 70 MB of disk, would be a better choice for long-term growth because of the larger fixed disk capacity growth path that is available with these machines.

#### **Standalone System with OS/2 Database Manager and DOS**

**Compatibility Feature:** Figure 4 illustrates the use of the OS/2 Extended Edition Database Manager in a standalone system.

The Database Manager is an integral component of OS/2 Extended Edition that supports the relational data model, in which data is externalized as a collection of tables. A table consists of rows and columns of data, and the user defines and accesses data in terms of these tables.

The Database Manager is consistent with IBM DB2, IBM SQL/DS, IBM SQL/400, and IBM's Systems Application Architecture. Data defini-

tion, retrieval, update, and control operations are supported by the Structured Query Language (SQL). SQL is available to this user interactively through the Query Manager, which provides a prompted interface designed to give easy-to-use access to the database.

A PS/2 Model 50 Z with 5 MB of memory and 30 MB of fixed disk would support this scenario.

*OS/2 will swap segments to disk in order to free real memory space.*

#### Communications Manager, APPC, and DOS Compatibility

**Feature:** Figure 5 adds the powerful Advanced Program-to-Program Communications (APPC) facility of OS/2 Extended Edition to the scenario shown in Figure 3.

With APPC, this user has the capability to draw on function that is partially resident on another system in a distributed manner. At the same time, a 3270 host communications session is active, as well as a word processing application, a spreadsheet, and a DOS application.

With five applications concurrently active, and the high potential for further growth for this user, a PS/2 Model 60-041 with 6 MB of memory and 44 MB of fixed disk storage is recommended as the minimum system to satisfy this user's initial capacity requirements.

OS/2 Standard Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	7.6
Segment Swap Data Set	N/A	N/A	2.5	2.5
Word Processing (estimate)	0.4 to 0.9	0.4 to 0.9	2.0	2.0
Spreadsheet (estimate)	0.5	0.5	1.5 to 3.5	1.5 to 3.5
Application data file estimates:				
Documents	0.005	0.005	0.5	0.5
Spreadsheets	0.04	0.04	1.0	1.0
<b>Totals</b>	<b>2.445</b> to <b>2.945</b>	<b>2.945</b> to <b>3.445</b>	<b>11.4</b> to <b>13.4</b>	<b>15.7</b> to <b>17.7</b>

Figure 1. OS/2 End-User Scenario: Stand-alone, No DOS Compatibility

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	8.2
Segment Swap Data Set	N/A	N/A	2.5	2.5
Communications Manager Base Code	0.2	0.2	3.7	2.7
3270 Communications	0.3	0.3	0.9	1.0
File Transfer (Note 1)	0.3	0.3	N/A	N/A
Configuration Files	N/A	N/A	0.5	0.5
Word Processing (estimate)	0.4 to 0.9	0.4 to 0.9	2.0	2.0
Spreadsheet (estimate)	0.5	0.5	1.5 to 3.5	1.5 to 3.5
Application data file estimates:				
Documents	0.005	0.005	0.5	0.5
Spreadsheets	0.04	0.04	1.0	1.0
<b>Totals (Note 2)</b>	<b>3.245</b> to <b>3.745</b>	<b>3.745</b> to <b>4.245</b>	<b>16.5</b> to <b>18.5</b>	<b>19.9</b> to <b>21.9</b>

Figure 2. OS/2 End-User Scenario: Entry Level Case A, No DOS Compatibility

**Communications Manager, Database Manager, and DOS Compatibility Feature:** Figure 6 shows the use of the OS/2 Communications Manager, the OS/2 Database Manager, and the DOS compatibility Feature.

Figure 6 shows the type of workload that an advanced user may implement to take full advantage of OS/2 Extended Edition. In this scenario, six applications are concurrently active. In addition to the 3270 host session(s), the spreadsheet and word processing applications, and the DOS Compatibility application, this user is also taking advantage of the integrated relational Database Manager and its associated Query Manager facility. Also, the user has established communications with a dial-up data service facility using the Extended Edition's 3101 ASCII Terminal Emulator component.

The presence of the configuration services function indicates that this user is capable of defining new configuration files for communications, and may be responsible for distributing tailored profiles to other users in the installation. The addition of the programming interfaces in the system implies that this user also does some application development for the communications environment.

Clearly, this user is attempting to maximize productivity by processing a variety of work concurrently.

Because of the demands placed upon the workstation for capacity and performance by this workload, it is recommended that a system with high growth potential and high performance characteristics be used.

A PS/2 Model 70 386-E61 with 8 MB of memory and 60 MB of fixed

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	8.2
Segment Swap Data Set	N/A	N/A	2.5	2.5
DOS Compatibility Application (Note 3)	0.5	0.5	2.0 to 3.0	2.0 to 3.0
Communications Manager Base	0.2	0.2	3.7	2.7
3270 Communications	0.3	0.3	0.9	1.0
File Transfer (Note 1)	0.3	0.3	N/A	N/A
Configuration Files	N/A	N/A	0.5	0.5
Word Processing (estimate)	0.4 to 0.9	0.4 to 0.9	2.0	2.0
Spreadsheet (estimate)	0.5	0.5	1.5 to 3.5	1.5 to 3.5
Application data file estimates:				
Documents	0.005	0.005	0.5	0.5
Spreadsheets	0.04	0.04	1.0	1.0
<b>Totals (Note 2)</b>	<b>3.745</b>	<b>4.245</b>	<b>18.5</b>	<b>21.9</b>
	to <b>4.245</b>	to <b>4.745</b>	to <b>21.5</b>	to <b>24.9</b>

Figure 3. OS/2 End-User Scenario: Entry Level Case B, With DOS Compatibility

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	8.2
Segment Swap Data Set	N/A	N/A	2.5	2.5
DOS Compatibility Application (Note 3)	0.5	0.5	2.0 to 3.0	2.0 to 3.0
Database Services (Note 4)	1.0	1.0	4.1	4.1
Query Manager	1.0	1.0	4.4	4.4
Application data file estimates:				
Database Application Data	N/A	N/A	0.6	0.6
<b>Totals</b>	<b>4.0</b>	<b>4.5</b>	<b>17.5</b>	<b>21.8</b>
			to <b>18.5</b>	to <b>22.8</b>

Figure 4. OS/2 End-User Scenario: Stand-alone Database and DOS Compatibility

disk would be an appropriate selection for this user. Alternate choices would be a 10 MB PS/2 Model 80 386-041 with 44 MB of fixed disk storage, or a PS/2 Model 80 386-071 with 70 MB of fixed disk storage.

### Communications Manager Plus

**LAN Requestor:** Figure 7 shows a system that is being used as a node on a local area network.

Figure 7 shows an environment that is becoming more common as local area networks proliferate.

This user is making use of OS/2's multitasking capability by running a 3270 host session and two concurrent local applications (word processing and spreadsheet), and at the same time connecting to a LAN server. The LAN server connection gives this user access to a much broader base of application programs and data files than the capacity of the local system would permit.

In this way, economies of system cost can be realized, because the individual user's workstation does not have to contain the capacity that would be required to store and run the entire range of applications that are available. Instead, a single large LAN server can satisfy a major portion of the capacity requirements of a collection of end users who have relatively small local configurations.

User requirements for this scenario could be satisfied by a PS/2 Model 50 or 50 Z with 4 MB to 5 MB of memory and from 20 MB to 30 MB of fixed disk storage.

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	8.2
Segment Swap Data Set	N/A	N/A	2.5	2.5
DOS Compatibility Application (Note 3)	0.5	0.5	2.0 to 3.0	2.0 to 3.0
Communications Manager Base Code	0.2	0.2	3.7	2.7
3270 Communications	0.3	0.3	0.9	1.0
File Transfer (Note 1)	0.3	0.3	N/A	N/A
Configuration Files	N/A	N/A	0.5	0.5
APPC (LU 6.2) Base (Note 5)	0.4	0.4	N/A	N/A
Application (estimate)	0.6	0.6	2.0	2.0
Data (estimate)	0.005	0.005	0.1	0.1
Word Processing (estimate)	0.4 to 0.9	0.4 to 0.9	2.0	2.0
Spreadsheet (estimate)	0.5	0.5	1.5 to 3.5	1.5 to 3.5
Application data file estimates:				
Documents	0.005	0.005	0.5	0.5
Spreadsheets	0.04	0.04	1.0	1.0
<b>Totals (Note 2)</b>	<b>4.75</b>	<b>5.25</b>	<b>20.6</b>	<b>24.0</b>
	to <b>5.25</b>	to <b>5.75</b>	to <b>23.6</b>	to <b>27.0</b>

Figure 5. OS/2 End-User Scenario: Medium-Load Environment

### Scenario Conclusions

As the preceding scenarios illustrate, careful planning is required to select the correct system configuration for OS/2 to satisfy both short-term and long-term requirements. Each installation that engages in capacity planning for system acquisition will have requirements that are unique to its characteristic set of operating environments. A detailed analysis of short-term capacity requirements, projections of growth as applications migrate to the OS/2 base, and considerations of anticipated new

applications will pay large dividends by helping to avoid the need to institute disruptive change in the future.

IBM's PS/2 product line is designed to satisfy the full range of your installation's capacity requirements over the long term.

### Capacity Planning Worksheets

Two capacity planning worksheets are furnished below. The first one



(Figure 8) provides suggested steps for estimating the memory requirements for OS/2 functions. The second worksheet (Figure 9) provides, in similar format, the steps necessary to estimate fixed disk size for OS/2 functions.

The worksheets can be used to estimate the capacity requirements for OS/2 Standard Edition Versions 1.0 and 1.1 as well as OS/2 Extended Edition Versions 1.0 and 1.1.

The memory estimates given are for the concurrent operation of the customer-selected OS/2 tasks using installation defaults, and are designed to provide a balance between performance and memory. All estimates are additive. By following the worksheet format and adding your applications and user data requirements, your memory and fixed disk storage estimates can be determined. Changes to the installation defaults such as the inclusion of a virtual disk (VDISK) or increased disk cache size will increase memory requirements.

**Memory Requirements:** Figure 8 shows the amounts of memory needed by the components of OS/2.

The numbers shown in Figure 8 are the recommended memory size in megabytes for concurrent operation of designated OS/2 functions. These numbers reflect the working set of the OS/2 functions listed, and not the total size of all the code and data areas that comprise the functions. The quantities shown assume that an acceptable degree of segment swapping is occurring in the operational environment. Refer to the "Interpretation of Memory Requirements" section for a description of the methodology used to develop these quantities.

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	1.5	2.0	3.9	8.2
Segment Swap Data Set	N/A	N/A	5.0	5.0
DOS Compatibility Application (Note 3)	0.5	0.5	2.0 to 3.0	2.0 to 3.0
Communications Manager Base Code	0.2	0.2	3.7	2.7
3270 Communications	0.3	0.3	0.9	1.0
File Transfer (Note 1)	0.3	0.3	N/A	N/A
3101 ASCII Terminal Emulation	0.3	0.3	0.1	0.3
Configuration Files	N/A	N/A	0.5	0.5
Configuration Services	N/A	N/A	N/A	1.5
Programming Interfaces	N/A	N/A	N/A	0.3
Database Services (Note 4)	1.0	1.0	4.1	4.1
Query Manager	1.0	1.0	4.4	4.4
Word Processing (estimate)	0.4 to 0.9	0.4 to 0.9	2.0	2.0
Spreadsheet (estimate)	0.5	0.5	1.5 to 3.5	1.5 to 3.5
Application data file estimates:				
Database Application Data	N/A	N/A	0.6	0.6
Documents	0.005	0.005	0.5	0.5
Spreadsheets	0.04	0.04	1.0	1.0
<b>Totals</b>	<b>6.045</b> to <b>6.545</b>	<b>6.545</b> to <b>7.045</b>	<b>30.2</b> to <b>33.2</b>	<b>34.6</b> to <b>38.6</b>

Figure 6. OS/2 End-User Scenario: Heavy-Load Environment

To estimate memory requirements, add together the recommended memory size for the operating system base and each installed function and application, and an allowance for user data buffer requirements.

If the user does not wish to concurrently execute all installed functions and applications, then the memory required will be less. In this case, the required memory size would be

the sum of the Operating System Base and the particular functions and applications the user wishes to execute concurrently, plus an allowance for user data. Using less than the recommended memory size will increase the time to switch from one application to another, reduce throughput, and increase the size of the swap file (see Figure 9).

### Fixed Disk Storage Requirements:

Figure 9 shows the fixed disk storage requirements of the components of OS/2.

### DOS Compatibility Mode Structure

When using Operating System/2, there is an option to use a DOS Compatibility Mode. This allows a user to continue using DOS-based software under OS/2.

Although a high percentage of DOS programs have been found to run successfully in DOS Compatibility Mode, there are some restrictions which, if violated, will prevent successful execution of some programs in this mode. For example, timing-dependent programs, a class of programs that includes DOS communications software, will not run reliably in this mode. For a complete list of restrictions, see the *OS/2 Technical Reference* manual.

The base system is installed with the minimum configuration and several default device drivers. The default drivers include the keyboard, display, clock, disks and printer. This configuration also provides parallel and serial ports.

The minimum default configuration for OS/2 Version 1.0 uses 130 KB in the low fixed portion of the system. Assuming that the CONFIG.SYS parameter RMSIZE = 640 (default) was used, this leaves 510 KB of storage for DOS program applications. Installation of additional drivers will further reduce the amount of DOS space available for use by applications.

Based on the actual configuration you have installed, this number will vary. To determine the actual amount of memory available in a

OS/2 Extended Edition Version	Memory (MBytes) (Working Set)		Fixed Disk (MBytes)	
	V1.0	V1.1	V1.0	V1.1
Base Operating System with Spooler	N/A	2.0	N/A	8.2
Segment Swap Data Set	N/A	N/A	N/A	2.5
Communications Manager (Note 6)	N/A	0.2	N/A	2.7
3270 Communications	N/A	0.3	N/A	1.0
LAN Requestor	N/A	0.3	N/A	4.9
Configuration Files	N/A	N/A	0.5	0.5
Word Processing (estimate)	N/A	0.4 to 0.9	N/A	(Server)
Spreadsheet (estimate)	N/A	0.5	N/A	(Server)
Application data file estimates:				
Documents	N/A	0.005	N/A	(Server)
Spreadsheets	N/A	0.04	N/A	(Server)
<b>Totals (Note 2)</b>	<b>N/A</b>	<b>3.745</b> to <b>4.245</b>	<b>N/A</b>	<b>19.8</b>

Figure 7. OS/2 End-User Scenario: LAN Requestor

given environment, the CHKDSK function can be used. After the installation has been completed and the system has been initialized, verify that you are in DOS Mode. When this has been completed, enter the CHKDSK command. (From the DOS prompt, enter CHKDSK.) This will display the amount of fixed disk space used and available, and the amount of DOS compatibility mode memory available for applications.

Figure 10 shows a sample CHKDSK report.

OS/2 Extended Edition supplies additional device drivers to support Communications Manager and Database Manager components. These device drivers are placed in low memory when selected, therefore reducing the memory available for the DOS application.

The drivers supplied with OS/2 Extended Edition, in addition to the ones contained in OS/2 Standard Edition, are shown in Figure 11.

### Virtual Storage and Segment Swapping

Operating System/2 has the ability to supply more usable memory than the amount of memory physically installed in the computer. This is accomplished by virtual memory management, using a technique called segment swapping.

Segment swapping becomes necessary when the user loads a combination of applications that collectively require more memory than the amount physically installed. This situation is called memory overcommitment. In this case, OS/2 will swap segments to disk in order to

free real memory space for additional applications.

For swapping to occur, the system requires enough free disk storage to contain the maximum number of segments that will be swapped to disk during an OS/2 session, that is, from the time that OS/2 is booted until it is rebooted.

The size of the swap file at system initialization is 640 KB. If the system determines that additional space is required, it is dynamically allocated when needed.

For example, if a system contains 3 MB of physical memory, but the required memory – the sum of system space, application program space, and data space being used at a specific time – is 4.5 MB, then the amount of overcommitted memory is 1.5 MB. In this case, the swap data set expands to approximately 1.5 MB during system operation, and it does not reduce in size until the user reboots OS/2.

In order to ensure peak performance of a system, it is desirable to minimize the amount of swapping that may occur. This can be done by providing as much physical memory as will be used on a regular and consistent basis.

To ensure that there will always be some free space on the fixed disk, you can use a parameter called MINFREE, which is available in Version 1.1 of OS/2 SE and EE. This parameter is specified in the SWAPPATH statement of the CONFIG.SYS file. MINFREE limits the growth of the swap file and ensures that the user will always have at least the minimum amount of free space on the fixed disk.

OS/2 Memory Requirements			
Functions	Component Size	Version	
		1.0	1.1
Operating System:			
Base (include for all estimates)		1.5	2.0
DOS Compatibility (Note A)	0.5	—	—
Communications Manager:			
Base (required for communications operations)	0.2	—	—
3270 Terminal Emulation	0.3	—	—
Asynchronous Terminal Emulation	0.3	—	—
APPC	0.4	—	—
File Transfer (Note B)	0.3	—	—
Database Manager:			
Database Services (required for database operations)	1.0	—	—
Query Manager	1.0	—	—
Local Area Network (Version 1.1 only):			
Requestor (Note C)	0.5	—	—
Server (IBM OS/2 Local Area Network Server V1.0) (Note C)	2.2	—	—
Application Programs:	Note D	—	—
Application Data Files:	Note E	—	—
Total		—	—
<b>Notes:</b>			
N/A - not applicable			
A - Automatically installed but removable by changing the PROTECTONLY statement in the CONFIG.SYS file			
B - Concurrent file transfer function will require 0.3 MB for each session			
C - Includes Communications Manager base code			
D - Provided by the application supplier			
E - Application- and user-dependent			

Figure 8. OS/2 Extended Edition Worksheet: Memory Size (in MB)

## Dynamic Link Libraries

Most of the OS/2 system-supplied support routines for applications are contained in the new OS/2 Dynamic Link Libraries. These operate in a very different way than similar routines in the DOS system. In DOS, system support routines were duplicated in every application program that needed them, as a result of the linkage edit process. In OS/2, a given routine that is needed by an application will most often be located and connected to the application at program load time or application execution time.

Furthermore, in a multitasking environment, if more than one application thread (that is, task) requires a support routine at the same time, OS/2 will share the same physical copy of the memory-resident routine among them.

This feature of OS/2 changes the method of calculating storage requirements substantially from the method used for DOS. With OS/2, it is no longer necessary to add up the storage required for multiple copies of support routines as you would do for DOS or a DOS extension. Instead, the amount of storage required by an OS/2 routine that is contained in a Dynamic Link Library is accounted for only once, because OS/2 shares the single copy across applications.

## Working Sets

The segment-swapping algorithms of OS/2 tend to keep only the active segments of application programs in real memory, while inactive segments tend to be written to the swap data set during heavy storage over-commitment situations.

<b>Functions</b>	<b>Version</b>		<b>Version</b>	
	<b>1.0</b>	<b>1.1</b>	<b>1.0</b>	<b>1.1</b>
<b>Operating System:</b>				
Base (include for all estimates)	3.9	8.2	3.9	8.2
Swap File (Note A)	2.5	2.5	—	—
<b>Communications Manager (Note B):</b>				
Base (required for communications operations)	3.7	2.7	—	—
3270 Terminal Emulation, SRPI, APPC	0.9	1.0	—	—
Asynchronous Communications	0.1	0.3	—	—
Configuration Files	0.5	0.5	—	—
Configuration Services (Note C)	N/A	1.5	—	—
Programming Interfaces (Note C)	N/A	0.3	—	—
<b>Database Manager:</b>				
Database Services (required for database operations) (Note D)	4.1	4.1	—	—
Query Manager	4.4	4.4	—	—
<b>Local Area Network (Version 1.1 only) (Note B)</b>				
Requestor	N/A	4.9	N/A	—
Server (IBM OS/2 Local Area Network Server 1.0)	N/A	6.2	N/A	—
Application Programs:	Note E		—	—
Application Data Files:	Note F		—	—

Figure 9. OS/2 Extended Edition Worksheet: Fixed Disk Size in MB (Part 1 of 2)

The active segments that remain in real memory are called the "working set" of the program, and repre-

sent the actual real memory requirements of the application during execution.

Working sets tend to be appreciably smaller than the total size of program code. Therefore, you may consider the total size of a program's code and data requirements to be a worst-case estimate for capacity planning.

True storage requirements for a given application can be determined only by observing the system during operation.

**Notes:**

- N/A - not applicable
- A - Experience and testing indicate that a 2.5 MB swap area would be adequate for many OS/2 Versions 1.0 and 1.1 environments. However, because the swap area is dynamically allocated by the operating system, and is therefore subject to a number of variables as applications attempt to use more memory than is available, a swap file greater or less than that set forth above may be necessary.
- B - The Communications Manager is a prerequisite for the LAN Requestor and the OS/2 LAN Server.
- C - In OS/2 Extended Edition Version 1.1, Configuration Services and Programming Interfaces can be removed after system installation. For details, see the product documentation.
- D - Included is working space and system data for one database. Add 0.5 MB for each additional user database. This space is subject to a number of variables and does not include user data. Fixed disk requirements greater or less than that shown may be needed.
- E - Provided by the application supplier
- F - Application- and user-dependent

Figure 9. OS/2 Extended Edition Worksheet: Fixed Disk Size in MB (Part 2 of 2)

<u>Device Driver File Name</u>	<u>Function</u>	<u>Approximate Size in Bytes</u>
DFTDD.SYS	3270 DFT Communications Driver	1600
SDLCDD.SYS	SDLC Communications Driver	27000
ASYNCCDDA.SYS	ASYN for ACDI (AT or XT 286)	24000
ASYNCCDDDB.SYS	ASYN for ACDI (PS/2)	31000
TRNETDD.SYS	Token-Ring Driver (EE 1.1 Only)	96000
PCNETDD.SYS	PC Network Driver (EE 1.1 Only)	110000

Figure 11. OS/2 Extended Edition Device Drivers

```

21317632 bytes total disk space
 245760 bytes in 3 hidden files
 120832 bytes in 56 directories
17274880 bytes in 1119 user files
 3676160 bytes available on disk

( DOS mode storage report )
 655328 bytes total storage
 533136 bytes free
    
```

Figure 10. CHKDSK Listing Including DOS Compatibility Memory OS/2 Standard Edition V1.1

### Carryover of DOS Application Inventory

When planning for memory and fixed disk storage requirements for OS/2, remember to factor in the amount of capacity required for the storage and execution of previously installed DOS programs.

Because of the availability of the DOS Compatibility feature in OS/2, it is to be expected that users will require space for their existing DOS programs.

It is recommended that you study these requirements to ensure that sufficient memory and fixed disk storage are supplied for both OS/2 and its applications, as well as for DOS carryover applications.

# Optimizing Your Use of the DOS 4.00 Shell

Jennifer Wilson  
IBM Corporation  
Boca Raton, Florida

IBM DOS Version 4.00 takes a giant step toward making it easier for you to use your personal computer. This breakthrough is made possible by the easy-to-use Shell that is included with the DOS 4.00 operating system.

DOS Version 4.00 moves away from the DOS prompt (A>) of previous versions, which required that you know (or look up in books) all the capabilities and command syntaxes of the operating system. With its full-screen menu system, the DOS Version 4.00 Shell gives you easy access to programs that you can customize in your Start Programs menu. The Shell also gives you a picture of the contents of your machine, and it enables you to perform commands in the File System without having to know command syntax.

You can also access the command prompt for the less frequently used commands. The command prompt can be reached from the Main Group in Start Programs or by using a hot key from anywhere in the Shell.

The Shell gives you many options. You can run the Shell in either text mode or graphics mode. In both modes, you can use the mouse and keyboard either concurrently or separately as desired. Two different



memory configuration choices give you the best memory usage and performance for each setup.

As the IBM single-tasking operating system, DOS 4.00 provides features for a turnkey setup for new users, as well as many exciting features for the experienced user. The Shell is easily customized for groups in large enterprises and easily learned by a novice. Experienced users will find many options in the File System useful in their technical work, such as a file view in hexadecimal.

---

## *The DOS Shell gives you easy access to programs.*

DOS 4.00 supports the Personal System/2 and Personal Computer families with a minimum machine configuration of 256 KB. Running the Shell with all available options requires 360 KB. When running the Shell optimally, only 3.7 KB

remains resident, so that many applications can take advantage of the additional memory.

## Common User Access Design

The Shell's design uses many of the features of Common User Access (CUA), the user interface component of Systems Application Architecture (SAA), IBM's standard of design across systems. By using this interface, the DOS Shell has a similar look and feel to the OS/2 Presentation Manager.

Many of the CUA structural features can be seen in the main panel of the Shell, shown in Figure 1. Areas of the screen are indicated in capital letters.

**Note:** While the DOS Shell has both text and graphics modes, the figures shown use text mode.

Other CUA features relate to how the Shell is operated. The Shell is task-oriented, using the principle of object-action, wherein an object(s) is picked from the panel area, and then an action to be performed on the object(s) is chosen from the action bar pull-down. When functions in the action bar are unavailable, the items are de-emphasized.

Arrow keys move the cursor in lists, and F10 toggles the cursor between the action bar and the panel area. The actions listed in the Function Key Area may be activated using the mouse pointer or the keyboard.

The graphics-mode Shell uses icons and scroll bars. Scroll bars give mouse users access to scrollable lists and a visible indication of their position in the list.

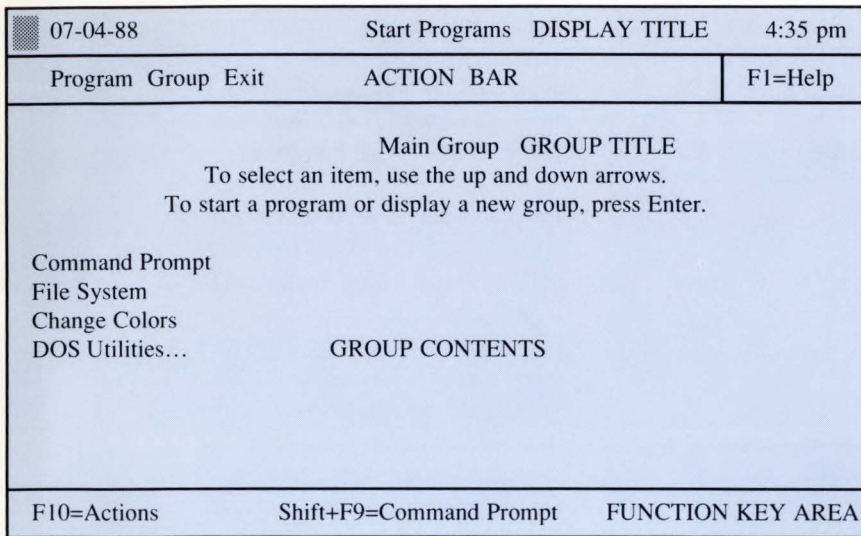


Figure 1. Start Programs Main Group

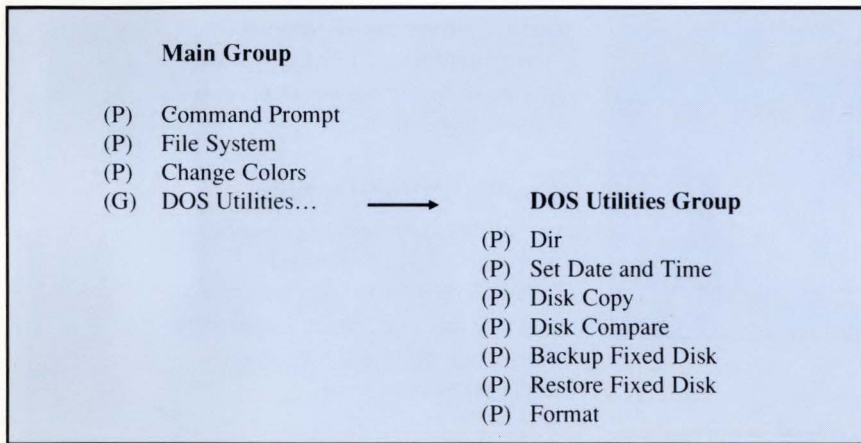


Figure 2. Start Program Group and Program Layout (P-program, G-group)

Contextual online help is available throughout the Shell. When help is requested by pressing the F1 key or by clicking the mouse pointer on F1=Help, users receive information about the item at the current cursor location. Both an index of help on subjects and a help that displays all key functions are also available.

## Shell Customizing

The Shell can easily be customized, either for yourself or to set up for others' use. One system can be set

up for many different individuals by using different groups (lists of programs) for each individual or by setting up across a network.

### Start Programs

Start Programs is the menu-driven part of the Shell that provides the ability for personalized groups and programs. Programs and groups can be given names of your choice.

The Main Group (see Figure 1), the first menu displayed, starts with several items available. The Main

Group can display names of programs, such as the File System or Change Colors, or names of groups, such as DOS Utilities. Colors may be customized by choosing from predefined palettes in the Change Colors program. The DOS Utilities group contains DOS functions that have already been customized for use (see Figure 2.) The ellipsis (...) following a title in group lists indicates that the item is a group. Programs may be started or groups displayed by selecting the title. When a group name is selected, the new group is displayed.

Both program and group names and group contents can be customized. Thus, when setting up your system, you may want to group your most frequently used programs into logical sections and create group names for them in the Main Group. A maximum of 16 programs and groups can be in the Main Group. Only programs can be added to groups other than the Main Group. Programs can also be moved to different groups, and both programs and groups can be reordered, added, and changed. See Figure 3 for the functions available in the Start Programs action bar. The ellipsis (...) in pull-downs indicates that a pop-up panel will follow.

*The Shell can easily be customized for use.*

Start Programs was designed to allow one person to easily set up a system for other users. When adding a program or group, the title, help information, password and filename (group) or commands (program) may be specified. See

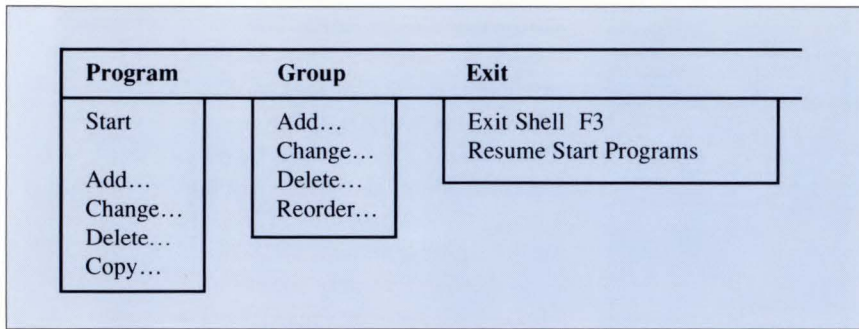


Figure 3. Action Bar Actions for Start Programs

Add Program	
Required	
Title . . . . . [	>
Commands . . . [	>
Optional	
Help text . . . . [	>
Password . . . . [       ]	
Esc=Cancel    F1=Help    F2=Save	

Figure 4. Add Program Pop-Up Window

Figure 4 for the pop-up window displayed when adding a program.

The ability to write your own help for each program and group is advantageous when you set up a system for others. The person desiring help on the item will see the help that was added when the item was created. Help can also be changed as desired. Help can be especially useful for recalling the syntax of commands or programs.

If a system is being set up for a number of different users who will have their own programs to run, passwords can be associated with groups or with programs.

Groups that have been set up by one user may be easily transferred to another user for use with DOS Version 4.00. When a group is added, a file with extension .MEU is created. After the programs (with help information and passwords as desired) have been added to the group, the filename given to the group can be copied to the new machine. To add the group to the new system, use the filename of the transferred file as the group filename in the Add Group panel.

Programs can actually perform a series of tasks using a batch-type language of program startup commands (PSCs). Multiple DOS com-

mands and applications can be executed. The PSCs must be separated by pressing F4 to insert the || character. Batch files must be run with the CALL command.

Programs can also be set up to request more information from the person who selected the program. A pop-up window requesting further information can be displayed when a program is selected. The pop-up can be customized for the user by defining program startup command parameters. See Figure 5 for parameters that may be used. Figures 6 and 7 show a set of program startup commands and the resulting display when the program is selected. The ECHO and PAUSE commands are useful in program startup commands to display information or to see the result of commands, respectively.

### Configuration Options

For further customizing, installation parameters can be changed. Parameters are also important for memory and performance considerations when running DOS on different types of systems.

### *Groups can be set up for each workstation.*

The Select installation program sets up, in the AUTOEXEC.BAT file, a batch file that starts the Shell. This file, DOSSHELL.BAT, invokes the two programs that run the Shell, SHELLB.COM and SHELLC.EXE. SHELLB is the 3.7 KB base Shell driver that remains resident in memory while the Shell is active. The parameters to customize the Shell are listed after SHELLC.EXE in DOSSHELL.BAT. SHELLC



[ ]	- display default popup prompt
[/T" title"]	- use "title" as title of popup prompt
[/I" instruction"]	- use "instruction" as instruction for popup prompt
[/P" prompt"]	- use "prompt" for entry field prompt
[%n]	- save the value which is entered for future use
%n	- use the value previously stored for variable n
[/D" default"]	- use "default" as the default for the entry field
[/D" %n"]	- display as the default for the entry field the value previously stored for variable n
[/R]	- clear the default value in the entry field when a key is pressed
[/L" n"]	- set the maximum length of the entry field to n
[/M" e"]	- do not allow entries which are not existing filenames
[/C" %n"]	- save the value previously entered for this variable as the current value
[/F" file"]	- check to make sure that "file" exists as a file before continuing
/#	- substitute the drive letter and colon
/@	- substitute the path from the root of the Shell directory

Figure 5. Program Startup Command Parameters

```

COPY C:\[%1 /T"EDIT AND SAVE BACKUP" /P"Filename . . "
      /I"Enter the name of the file to be edited."
      /D"AUTOEXEC.BAT"/L"12"] BACKUP ||
PE2 %1 ||

```

Figure 6. Menu Item Program Startup Commands

loads the functions specified by the parameters and displays the Start Programs screen. The size of SHELLC varies depending on the parameters included.

The parameters MENU, COLOR, DOS, PROMPT, EXIT and MAINT are useful for a person setting up a system for other people. These options can be removed to keep a person from accessing these functions. For example, in the education environment, programs can be set up for the students' use in the Start Programs menu. By configuring the Shell using the /MENU parameter but not the COLOR, DOS, PROMPT, EXIT and MAINT

parameters, students will be able to run the programs, but not change the color or have access to the files in the system.

To change the filename to be used for the Main Group, the MEU:filename parameter should be changed. This may be desirable when sharing menus among users. The mouse driver and color defaults are set by the MOS:filename and CLR:filename parameters, respectively. Several mouse drivers are included with the Shell for the IBM Mouse and the Microsoft serial and parallel mouse devices. To use a mouse device driver that implements the Microsoft Int 33 API, in-

clude the driver in one of the startup files (AUTOEXEC.BAT or CONFIG.SYS), and do not include the /MOS parameter.

The TEXT parameter is added to run the Shell in text mode, while CO1, CO2, and CO3 are the graphics-mode parameters to use to set modes 10, 11 and 12 respectively. (See Figure 8.) If none of these options is included, the mode giving the best resolution for the system will be the default.

Terminate-and-stay-resident (TSR) programs with a hot key may be used directly from the text mode Shell. However, in the graphics-mode Shell, you should first go to the command prompt (Shift+F9), then use the hot key to your TSR program.

## Memory Considerations

For best performance and program workspace, it is important to have the correct memory configuration for your machine setup. Transient mode, using the /TRAN parameter, and resident mode are the two options. In general, transient mode should be used when running DOS from a fixed disk or a virtual disk, and resident mode when running on a diskette-based system to keep diskette swapping to a minimum.

In resident mode the Shell returns more quickly after running a program. However, in transient mode more memory is available to run programs from the Shell. In transient mode, when a program is run, SHELLC.EXE is temporarily released from the memory space so the program can use the memory. When the program terminates, the Shell is loaded from secondary memory. In resident mode, both SHELLB and SHELLC stay resi-

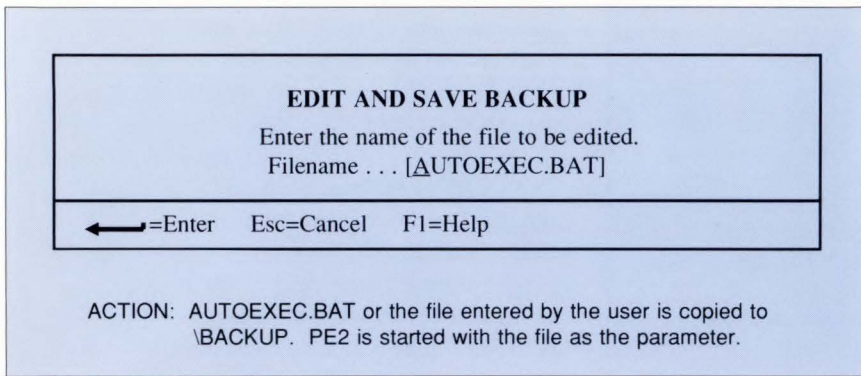


Figure 7. Display at Selection of Item

Shell parameter	Mode	Colors	Resolution
/CO1	10	16	640 x 350
/CO2	11	2	640 x 480
/CO3	12	16	640 x 480

Figure 8. Video Modes

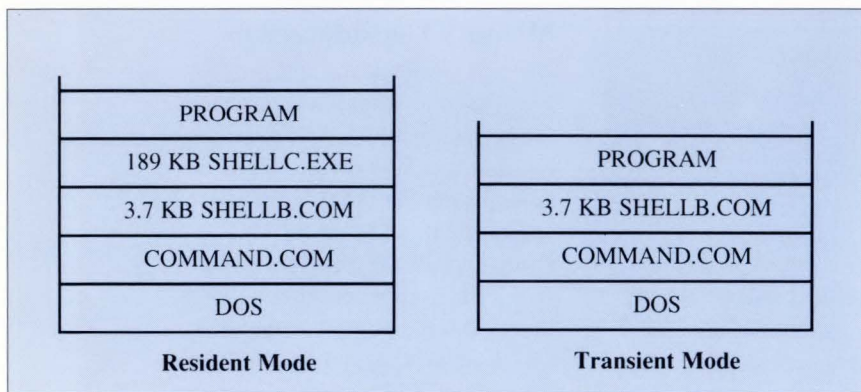


Figure 9. Memory Maps for Resident / Transient Modes When Running a Program from the Shell

dent, and the program is loaded in the next available memory. See the memory maps in Figure 9.

When running in resident mode, the /B:n parameter should be used to get the best use of memory. The n in this parameter is the number of kilobytes to be reserved for the File System buffer. This size can be approximately determined by the formula  $n = f * 57$  where f is the

number of files in the system. If the buffer is too small, an error message will be displayed, and not all files will be included in the file list. If this parameter is not used, the largest amount of space will be reserved.

The /SWAP parameter can be used to improve performance. When it is used, the File System buffer is loaded to a disk file while a

program or the command prompt is active.

When you start the Shell command prompt, another version of COMMAND.COM is loaded. To return to the Shell, type EXIT. If you exit the Shell by using the Exit pull-down or F3, you return to the DOS command prompt. Type DOSSHELL to return to the Shell. Figure 10 shows the memory maps of the Shell command prompt (Shell in transient mode) and the DOS command prompt.

TSR programs should be started before entering the Shell or from the Start Programs menu, rather than from the Shell command prompt.

The /MUL parameter allows you to use the multiple file list display in the File System. If you do not plan to use the multiple file list, do not set this option so that you can limit the memory reserved.

The default options set by Select are:

```
/MOS:PCIBMDRV.MOS/TRAN
/COLOR/DOS/MENU/MUL/SND
/MEU:SHELL.MEU
/CLR:SHELL.CLR/PROMPT
/MAINT/EXIT/SWAP/DATE
```

## Network Operation

The customizing features of the Shell are put to good use on a network. The Shell can be installed on the server, and groups for Start Programs can be set up on each workstation. Separate batch files can then be set up on each node so that the person at each node can have a customized system, all run from the same server.

Another option is to have all the batch files on the server, with different names for the batch, color,

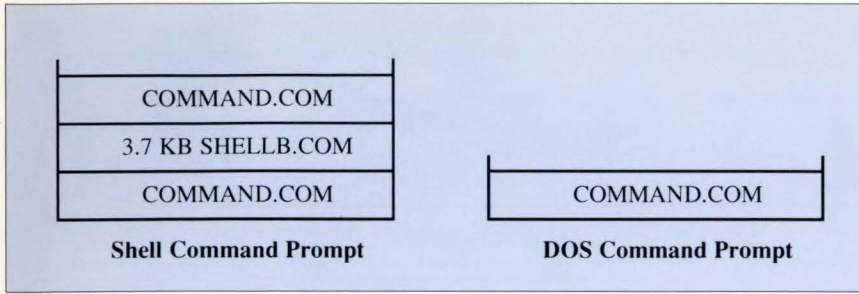


Figure 10. Memory Maps for Command Prompts

07-08-88	File System	DISPLAY TITLE	4:36 pm
File	Options	Arrange	Exit
ACTION BAR			F1=Help
A	B	C	D E F
DRIVE IDENTIFIER			
C:\			
PATH AREA			
Directory Tree	More: ↓	FILE LIST	**
C:\		BACKUP .COM	33,754 06-14-88
— WORDPF	DIRECTORY	BASIC .COM	1,065 06-14-88
— DOS	TREE	BASICA .COM	36,285 06-14-88
— DWS		CHKDSK .COM	17,771 06-14-88
— FDO		COMMAND .COM	37,637 06-14-88
— PE2		COMP .COM	9,491 06-14-88
— MNFRAME		CONFIG .SYS	162 07-08-88
— MYTE		COUNTRY .SYS	12,838 06-14-88
— OTHERS		DEBUG .COM	21,606 06-14-88
— ASSIST		DISKCOMP .COM	9,889 06-14-88
— PCSCMD		DISKCOPY .COM	10,428 06-14-88
— NET		DISPLAY .SYS	15,741 06-14-88
— TOOLS		DOSSHELL .BAT	200 07-08-88
— COMPILE		DOSSHELT .BAT	206 07-08-88
— MISC		DOSUTIL .MEU	7,696 07-08-88
F10=Actions	Shift+F9=Command Prompt	FUNCTION KEY AREA	

Figure 11. File System

and menu files for each node. If the same files are used for different people, they should be read-only, and the MAINT and COLOR options should be off.

## File System

The Shell File System gives a pictorial view of the drive, directory and file structure, and provides easy access to frequently used DOS functions and other features. Areas of the screen are shown in capital let-

ters in Figure 11. The directory tree structure and the files on the current directory for the default drive are displayed. In graphics mode the directory list and files can be viewed using a scroll bar. The path area shows the current drive and directory path.

The File System offers many functions from the action bar displayed in Figure 12. Many functions available in the File pull-down perform enhanced DOS commands on files

or directories. Several features that allow you to customize the File System for your use are available in the Options pull-down. The Arrange pull-down lets you select from three display formats.

## File Operations

Many of the commands in the File pull-down have special enhancements that were not available in previous versions of DOS. A view function allows the display of a file in either hexadecimal or ASCII. The ability to rename a directory containing files without copying the files to a new directory, and the ability to move a file without first copying and then deleting, are other enhanced functions. File attributes may be changed for a group of files or for each file individually. The ability to select all the files in the current directory, or to cancel selections, is also available.

*It is easy to run programs in the file system.*

To refresh the file list, the drive can be selected. The file list is also refreshed when returning from a program.

It is easy to run programs in the File System. With the keyboard, pressing Enter while on an executable file will start the program, as will double-clicking the mouse button while the mouse pointer is on the file. A third method is to select Open from the Program pull-down. Program parameters may be entered if desired.

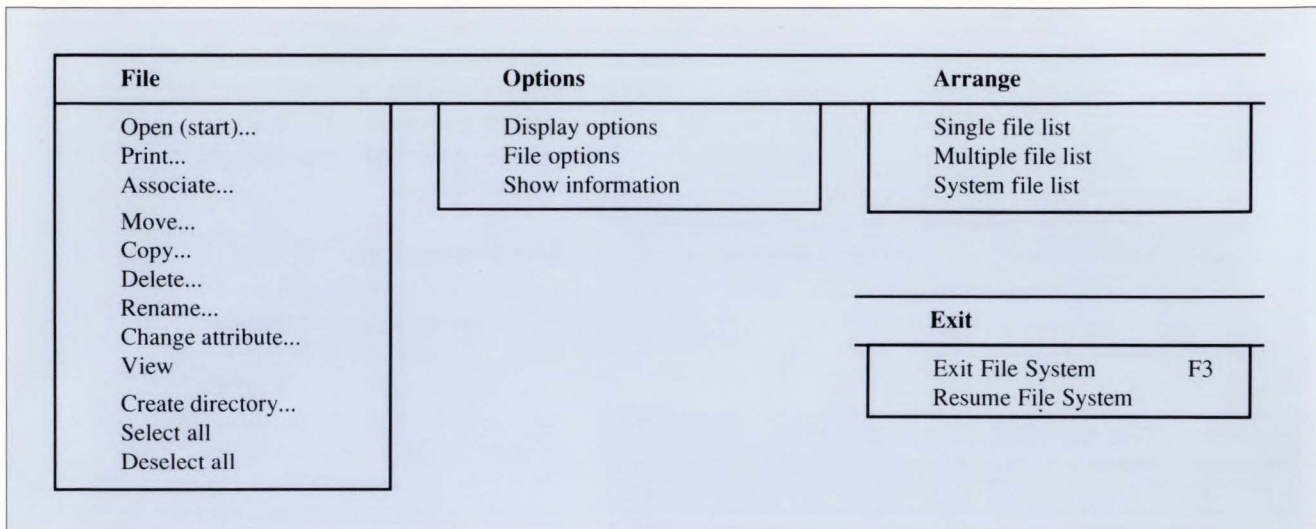


Figure 12. Action Bar Actions for File System

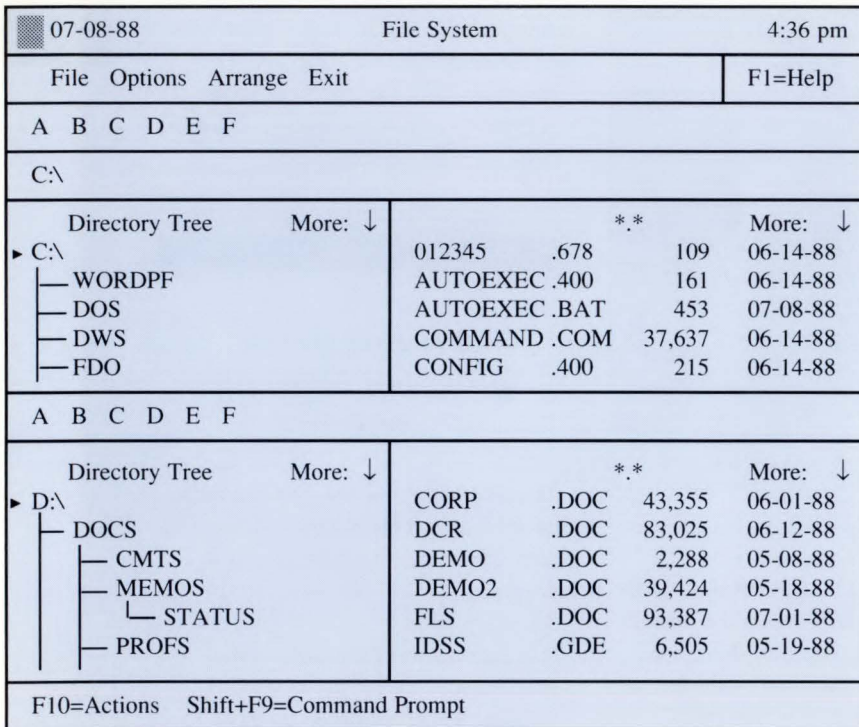


Figure 13. Multiple File List

An additional method is provided to run programs associated with files. Data files with a .DOC extension can be associated with a particular word processor, for example, so that if a .DOC file is selected to be run,

the word processor is started with the .DOC file as its object.

### Screen Formats

Three formats are available in the File System. The default File Sys-

tem display, the Single file list, shows a drive list, directory tree and file list. By changing the display to Multiple file list (Figure 13) from the Arrange pull-down, two file lists from different drives and directories can be displayed. The System file list (Figure 14) shows a list of all the files on the drive. This display can be used as a search facility to locate all instances of a particular file.

Information about selected files and the current file, directory and drive is also displayed in the System file list. This information is available from the other displays as well as by selecting Show information from the Options pull-down.

### Customizing the File System

Several customizing features should be utilized for optimal use of the File System.

The files to display from the current directory can be changed by selecting Display Options from the Options pull-down and entering a filespec that acts as a filter. Wildcards (\*,?) may be used or a complete filename can be given.

07-08-88		File System				4:36 pm
File Options Arrange Exit					F1=Help	
A B C D E F						
C:\DOS33						
File					*.*	More: ↑ ↓
Name :	CONFIG.SYS	COMMAND	.COM	37,637	06-14-88	10:00pm
Attr :	....	COMMIT	.315	1,228	03-12-88	8:19pm
Selected	C	COMMUN	.MNU	256	04-26-87	9:09am
Number :	0	COMP	.COM	9,491	06-14-88	10:00pm
Size :	0	COMP	.COM	9,491	06-14-88	10:00pm
Directory		COMPATS	.PIC	11,598	02-15-85	12:00am
Name :	DOS33	COMPRSS	.ASM	18,488	01-07-86	5:25pm
Size :	762,023	COMPUTER	.FNT	7,424	01-05-87	12:08am
Files :	53	CONDENSE	.DAT	27	04-28-87	1:43am
Disk		CONFIG	.12	12	01-01-80	12:02am
Name :	THIS ISIT!	CONFIG	.400	215	06-14-88	11:19am
Size :	21,317,632	CONFIG	.BAT	16	07-08-88	12:05am
Avail :	833,536	CONFIG	.SYS	336	07-08-88	2:53pm
Files :	1,248	CONFIG	.SYS	11	07-01-88	12:01am
Dirs :	47	CONFIG	.SYS	217	07-07-88	4:56pm
F10=Actions Shift+F9=Command Prompt						

Figure 14. System File List

All files except a filespec may be displayed by using a tilde before the filespec. The default filespec is \*.\* so that all files in the directory are displayed. The filespec remains until it is changed, and is shown above the file list in all displays.

By using a filename as the filter in the System file list, you can do a drive search for that file. As you move the cursor down the file list, the subdirectory of each location of the file is displayed.

Another option from Display Options in the Options pull-down is the file sort. The order of the file list may be selected by date, size, disk order, name, or extension.

By selecting Mark Across Directories after selecting File options from the Options pull-down, operations can be performed on files throughout the drive. The Show information panel is useful when using this option to see how many files are marked in each directory and to see the total size of the marked files. This is useful to know before performing a copy to a limited-size area, such as a diskette. When deleting, the number and location of files marked is a safeguard to keep from deleting more than you planned.

Another safeguard provided is the ability to Confirm on Delete and Confirm on Replace. These also result from the File option in the Op-

tions pull-down. If Confirm on Delete is chosen, each deletion will be confirmed before the file is removed. If Confirm on Replace is chosen, a copy or move which will replace a file at the destination is confirmed before the previous file is overwritten.

The ability to customize the Shell in many ways allows optimal use for individuals and easy setup for groups. The DOS Version 4.00 Shell allows knowledgeable users to rarely have to use a command prompt, while having fast, easy access to the information and functions they desire. Novice users have no need to learn the complexities of the system while having many desirable features available to them.

# DOS Shell Program Startup Commands

Faye Smith  
IBM Corporation  
Boca Raton, Florida

IBM DOS Version 4.00 includes several enhancements that make it much easier to use.

One enhancement is SELECT, a menu-driven installation aid. When you install DOS Version 4.00, the choices you make from the menus in the SELECT program determine your system's configuration.

Another enhancement in DOS 4.00 is the DOS Shell. A shell is an interface between the user and the operating system. The DOS Shell, much like SELECT, makes it possible to accomplish common tasks by making selections from a menu rather than entering commands and their syntax at a command prompt.

Unless you choose during the installation process not to install the Shell, the Shell is automatically installed. When you start your system with DOS 4.00, the Main Group screen appears (see Figure 1 in preceding article).

**Note:** There are two display modes in the Shell – text and graphics. Throughout this article, the Shell is shown in text mode.

The Shell comes with a few things already in its Main Group. Choosing **Command Prompt** from this group takes you to the DOS command prompt. **DOS Utilities...**

takes you to another group from which you can choose several common utilities offered in the Shell. Choosing **File System** takes you to a screen from which you can manipulate all your files. And you can also change the colors of the Shell from the Main Group.

*The Shell comes with a few things already in its main group.*

What you will most likely want to do with this Group is add the title of an application program that you are presently using. Then, when you turn on your system and are presented with this Shell screen, you can simply select that program, and it starts.

When you add program titles to a Group in the DOS Shell, you can use the Shell's program startup commands (PSCs) to define how those programs work. To add a program, select **Program** from the action bar.

**Note:** When you select any of the choices in the action bar, a "pull-down" is displayed. A pull-down is a list of additional choices specific to the action bar choice you selected. A "pop-up" is used to request additional information concerning a choice in a pull-down.

Select **Add** from the Program pull-down. You are then presented with the Add Programs pop-up.

The cursor is at the **Title...** entry field in the Add Programs pop-up. Type the title you want to appear in the group to identify this program.

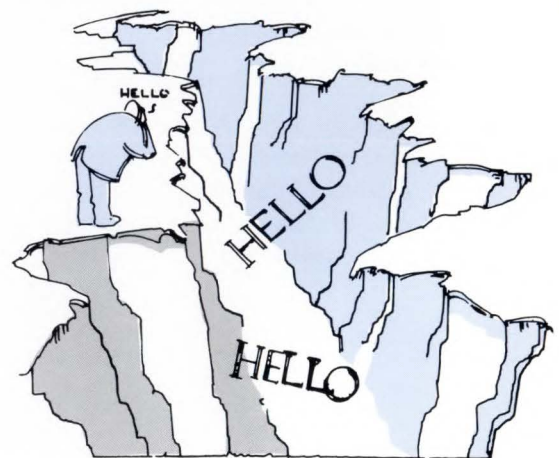
Then tab to the **Commands** entry field. It is in the **Commands** entry field of the pop-up that the PSCs are entered.

The simplest PSC consists of the characters you enter to start the program. These are identified in the documentation that comes with the program. When adding a program to the Shell, if you enter this command in the **Commands** entry field, the program starts when it is selected.

However, there are other, optional PSCs that you can use in the **Commands** entry field to further define the way the program works. For example, you can request one or more prompts for information each time the program is started. The default prompt is shown in Figure 1.

Each prompt panel consists of a title, an instruction line, and an entry field, and it is displayed when you start the program. The default prompt uses **Program Parameters** for the title; **Type the Parameters, then Press Enter** for the instruction line; and **Parameters..** to identify the entry field.

By using program startup commands offered in the Shell, the title, instruction line, and entry field



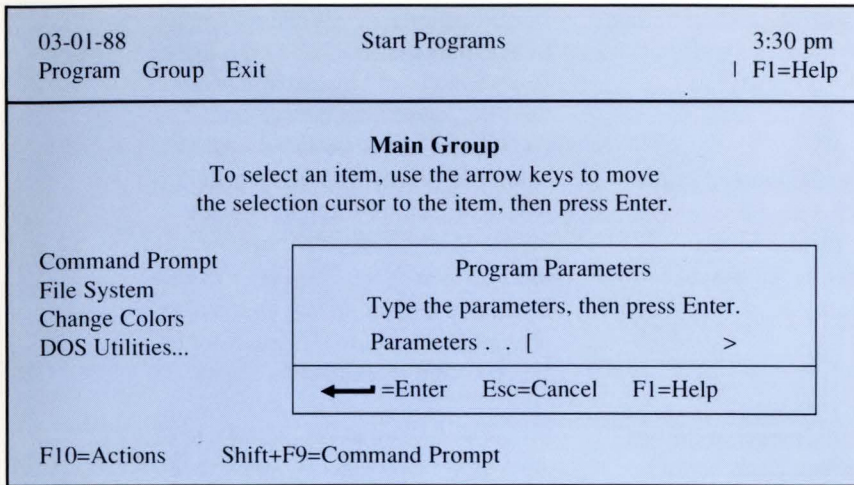


Figure 1.

prompt lines can have information that is specific to your needs. (The information entered in response to the prompt is passed as a parameter to the program you are starting.) In addition, you can tailor the activities of the program in other ways.

Following is a list of PSC options that the Shell offers. The options shown in brackets control the information in the prompt panel and must be entered between the brackets. Options shown between brackets can be combined inside one set of brackets. The options shown without brackets must be entered outside the brackets.

**Note:** In the examples that follow, the PSC strings may be shown on more than one line. This is done to show them in their entirety. On the screen, this is a scrollable field, and the commands are entered in a continuous string.

[ ]

Performs the default prompt each time the program is started. For example, if you enter:

```
EDITOR [ ]
```

in the **Commands** entry field, each time you select the Editor program, the default prompt will be displayed.

[/T".." ]

Defines a title for the prompt panel. The maximum length for the title is 40 characters.

[/I".." ]

Defines an instruction for the prompt panel. The maximum length for the instruction is 40 characters.

[/P".." ]

Defines a prompt for the entry field in the prompt panel. The maximum length for the prompt is 20 characters.

The following example shows how you could use the preceding three PSCs to define the prompt that will be displayed when you choose the Editor program.

```
EDITOR[/T"Editor"
/I"Enter name of file to be edited."
/P"Filename . . "]
```

[%n]

Saves what you have entered (the value) in the entry field for future use. The "n" can be any number from 1 through 10. When used, %n must be the first character or characters inside the brackets. See Example 2 later in this article.

%n

Uses the value entered at run time. For example, if the value is a file that has been edited, PRINT %1 causes the identified file to print each time this program is started. When used, this option must be entered outside the brackets. Refer to Example 2.

[/D".." ]

Makes a default value appear in the entry field each time a program's prompt is displayed. You can change this default value at run time by typing over it and clearing any remaining characters. This entry can be up to 40 characters long. For example:

```
EDITOR[/T"Editor"
/I"Enter name of file to be edited."
/P"Filename . . "
/D"MYFILE.SCR"]
```

[/D"%n" ]

Makes a default value appear in the entry field each time a program's prompt is displayed. This default is a previously entered value saved with the [%n] option. You can change this default value at run time by typing over it and clearing any remaining characters. This entry can be up to 40 characters long. See Examples 3 and 4 later in this article.

**[/R]**

Clears the default value in the entry field when the first key pressed is any key other than an edit key. In the example below, the **/R** PSC is added at the end of the string:

```
EDITOR[/T"Editor"
/I"Enter name of file to be edited."
/P"Filename . . ."
/D"MYFILE.SCR" /R]
```

With these PSCs entered, the prompt shown in Figure 2 is displayed when **Editor** is chosen from the group:

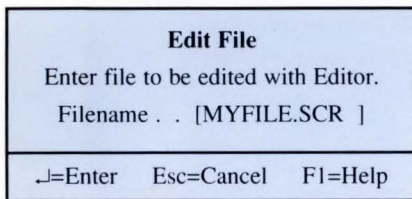


Figure 2.

Because the **/R** PSC was added, the default value (MYFILE.SCR) can be cleared by pressing any key other than an edit key.

**[/L"n"]**

Sets the maximum length of the entry field to fewer than 127 characters. The maximum length is 127 characters. If the length is not specified or is invalid, the maximum length is used. In the following example, only 12 characters can be entered in the entry field:

```
EDITOR[/T"Editor"
/I"Enter name of file to be edited."
/P"Filename . . ."
/L"12"]
```

**[/M"e"]**

Permits the use of only existing filenames. The existence of the

value entered will be verified before the PSC is executed.

**[/C" %n"]**

Saves the %n value entered in the preceding process as the value for this parameter for the current process; otherwise, %n has no value. See Example 3.

**[/F".." ]**

Checks for the existence of the file specified. This entry can be up to 76 characters long and can be used more than once. An example is `/F"d:\path\filename."` This check takes place after the Enter key has been pressed on the prompt panel, and is used to ensure that the proper diskette is in the drive before the Shell is suspended to execute the program startup commands. If the file exists, the PSC continues. If the file does not exist, a beep sounds and the panel is redisplayed. This parameter has no effect on the format of the prompt, and is not included in the program startup commands that are executed.

**/#**

Substitutes the drive letter (that designates the drive from which the Shell was started) and a colon into the PSC. These characters must be entered outside the brackets.

**/@**

Substitutes the path from the ROOT from which the Shell was started, including the current directory, into the PSC. The path is not preceded by a backslash. These characters must be entered outside the brackets.

*Note:* Any batch file command, with the exception of GOTO statements, can be used as a program

startup command. (When using the FOR command, the "n" in %n must be alphabetic.) Any characters in the PSC, other than the optional commands, are passed to the DOS batch file processor exactly as written. A direct substitution is made by the Shell for any PSC option used outside of the brackets. The information entered in response to the prompt is substituted for each set of brackets and its enclosed options.

Press the F4 key at the end of each command when entering more than one program startup command in the **Commands** entry field.

**Batch Files**

To start a batch file using a PSC, the batch file name must be preceded by CALL. If the batch file name is not preceded by CALL, the program will not return properly on completion. For example, to start a batch file named BATCH.BAT, you would enter the following PSC at the **Commands** entry field:

```
CALL BATCH.BAT
```

If you want to use some of the options offered for program startup commands, you must include the contents of the batch file in the PSC. Use the bracketed options to prompt for any required parameters.

**PAUSE and ECHO Commands**

It is useful to insert a PAUSE command at the end of each PSC list so that, before returning to the Shell from your program, you can view any DOS messages that may appear.

If you add an ECHO command with a message statement to your PSC list, that message appears on your screen. The ECHO message can be



whatever remark is correct for the program you are starting.

In the following example, the remark "Insert diskette with Editor in drive A" is used:

```
[ECHO Insert diskette with Editor>
in drive A||PAUSE||EDITOR
```

Note that the || is the symbol that appears when you press F4 to separate commands. By entering the ECHO command with this remark, the following is displayed when the editor program is selected:

```
Insert diskette with Editor in drive A
Press any key to continue . . .
```

*The ECHO message can be whatever remark is correct.*

At the ECHO command, you have the opportunity to insert the needed diskette; at the PAUSE command, the system pauses to display any DOS messages, and the system again displays the message "Press any key to continue..." before returning to the Shell.

### Using Additional Program Startup Commands

Following are some examples of ways to use program startup commands.

**Example 1.** If you add a program (for example, Editor) to one of your groups, you can enter the following

in the **Command** entry field to change the default prompt:

```
EDITOR [/T"Edit File"
/I"Enter file to be edited with
Editor."
/P"Filename . . ."
/L"12"]
```

By entering these program startup commands for the Editor program, when you choose **Editor** from the group, you see the prompt in Figure 3.

Edit File		
Enter file to be edited with Editor.		
Filename . . . [		]
␣=Enter	Esc=Cancel	F1=Help

Figure 3.

Now type the name of the file you want to edit in the entry field. This can be the filename of an existing file or a file you want to create. (For example, if you enter MYFILE.SCR at the prompt, the Editor program is started using MYFILE.SCR as a parameter.) Then press Enter.

**Example 2.** Any DOS command, except a GOTO command, can be entered as a PSC. For example, to add a PRINT command to the command list shown previously, the commands are entered as follows:

```
EDITOR [%1/T"Edit File"
/I"Enter file to be edited with
Editor."
/P"Filename . . ."
/L"12"]||
PRINT %1
```

The %1 causes the parameter entered in response to the prompt panel to be passed to the PRINT program. The PSC list entered in this way still gives you the prompt when the program is selected, and it sends the document to print before returning to the Shell.

**Example 3.** The /C"%n" PSC is used to retrieve a value that has been saved by the previous process using the %n PSC. For example, if you have an editor and a formatter in your group, you can edit a file using the editor. Then, immediately after exiting the editor, when you start the Formatter program, the %n value is retrieved and used in the entry field. The /D"%n" PSC is used to initialize an entry field with the value that was retrieved using the /C"%n" PSC. The PSCs used for the formatter are as follows:

```
FORMATTER [/C"%1"/D"%1"]
```

**Example 4.** By using the /D"%n" PSC, you can use a previously entered variable as a default for a second input field. For example, you can edit a file with an editor and, before returning to the Shell, send the file to a formatter. The program startup commands for the editor would be as follows:

```
EDITOR [%1/P"Edit File"]||
FORMATTER [/P"Script
File"/D"%1"]
```

The %1 saves the value of the entry field (the name of the file being edited). The /D"%1" PSC retrieves that information and uses it for the value in the second prompt to identify the file, which is then going to the formatter.

# Discover/ Education: Learning with a Difference

Saul Carliner  
IBM Corporation  
Atlanta, Georgia

One of the challenges to OS/2 users at all levels is learning how to effectively use the many functions and facilities of this operating system.

IBM's Discover/Education Series helps users meet this challenge. Through these computer-based, self-study courses, all types of users – from basic users to PC support people to technical professionals – can learn about OS/2 and its Database Manager. (Similar courses are available about DOS and for PC and PS/2 planning and installation.) Users install a course and take it on their own IBM Personal Computers and IBM Personal System/2 computers using DOS or OS/2 (ex-

cept IBM 3270 Personal Computer and IBM 3270 Personal Computer AT).

## What Discover/Education Courses Teach

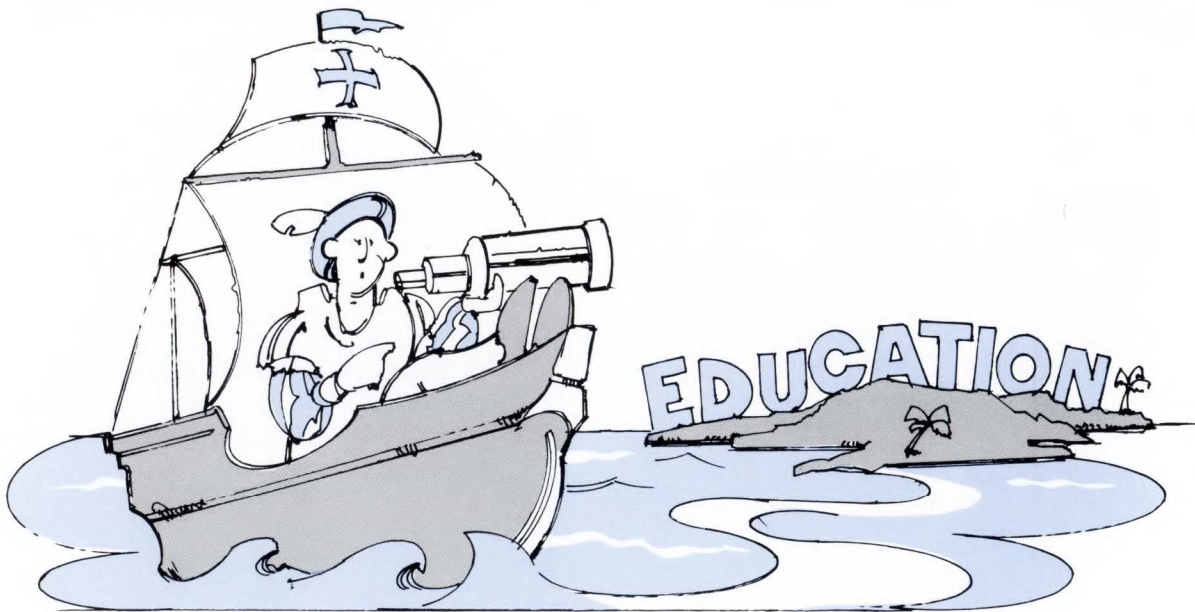
IBM offers two Discover/Education courses about OS/2. The first, *OS/2 Introduction and Operations*, teaches how to:

- Use OS/2 commands in both OS/2 and DOS modes.
- Use the functions available with the OS/2 Presentation Manager.
- Use the help facilities of OS/2.
- Respond to OS/2 messages.
- Use the OS/2 System Editor.
- Use OS/2 functions such as piping, filtering, redirection, and conditional processing.
- Create and modify STARTUP.CMD and AUTOEXEC.BAT files.

- Use OS/2 commands and functions that affect system performance.
- Interpret and modify statements in the CONFIG.SYS file.
- Identify the procedures to resolve possible OS/2 problems.
- Define terms associated with OS/2.
- Describe the internal structure and functions of OS/2 memory management, device management, and process management.

The second course, *OS/2 Database Manager Introduction and Operations*, teaches how to:

- Identify the purpose and advantages of a relational database.
- Plan for and create an efficient database.
- Install and configure OS/2 Database Manager.



- Use Query Manager to create tables, indexes, and views.
- Use Query Manager to select, retrieve, and edit data.
- Use Query Manager to customize a report in the desired format.
- Use Query Manager to build procedures and create customized menus and panels.
- Identify the procedures necessary to backup and restore a database.
- Identify the data protection services included with OS/2 Database Manager.
- Use SQL to edit and retrieve information from the database.

*Users are protected from errors while practicing new skills.*

### Personalized Training

As important as *what* Discover/Education courses teach is *how* the course material is taught. Discover/Education is designed to provide a personalized training experience that provides the right training for the user, at the user's convenience, on the user's own system.

Helping users to select the right training is an administration program that is included with the course. Just as OS/2 can provide either simple or sophisticated capabilities, depending on the needs of the user, so the sophisticated

IBM Discover/Education	
<b>Select an Audience Path</b>	
Course title . : OS/2 Introduction and Implementation	
Move cursor to desired audience path; type an option; press Enter. 1=Select 5=Display modules 8=Display description	
Option	Audience path title
-	Basic User
-	Advanced User
-	Support Person
-	All Modules in the Course
Enter F1= Help F3=Exit	

**Figure 1. Panel from Which Users Select an Audience Path**

software of Discover/Education can provide basic or in-depth training, again depending on the user's needs.

When users first enroll in a Discover/Education course, they identify the audience to which they belong, such as a basic user or advanced user. Figure 1 shows an example of the panel on which users identify their audience for *OS/2 Introduction and Implementation*.

Based on this designation, the administration program chooses a path of units, called *modules*, for users to follow through the course. Within these audience paths, users only take the modules they need. If desired, users can also take modules that are not in their audience paths.

The administration program acts like a registrar. In addition to enrolling users in appropriate modules, the administration program keeps track of users' progress through the course. A person in your organization who is designated as an administrator can

view the records about users' progress.

Once into the course, users learn by doing. OS/2 Discover/Education courses solicit a high degree of interaction to make education both a practical and pleasant experience.

In *Discover/Education OS/2 Introduction and Implementation*, users install OS/2, work with the Presentation Manager, enter commands, design a batch file, use the System Editor, customize a configuration file, and troubleshoot problems. These tasks are performed in a simulated environment, so users are protected from errors while practicing new skills.

In *Discover/Education OS/2 Database Manager Introduction and Implementation*, users learn how to use the Query Manager prompted interface, enter SQL statements, and configure the Database Manager. Figure 2 shows an example of the type of exercises that

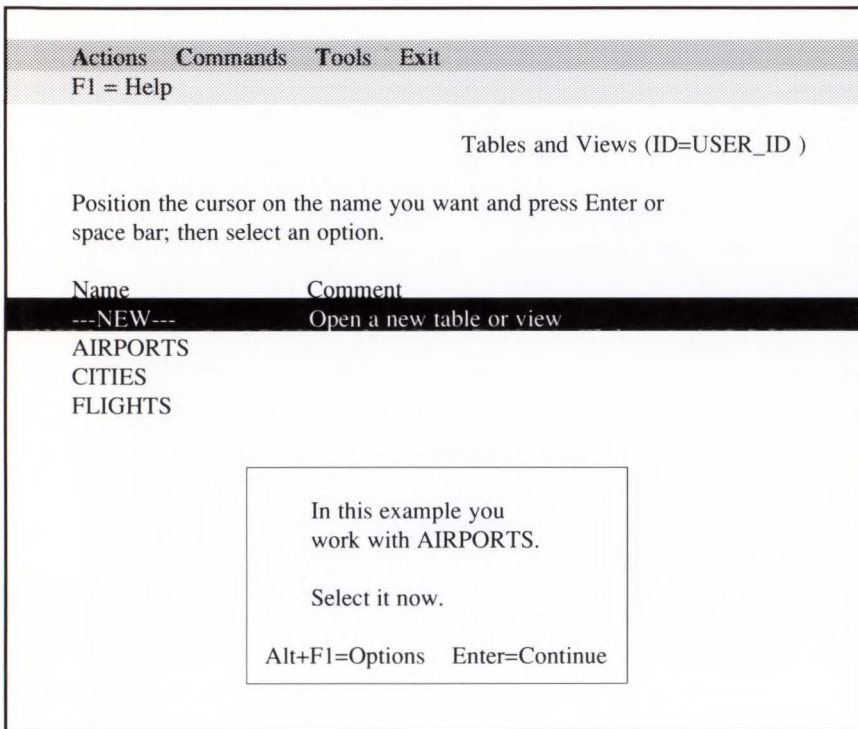


Figure 2. Example of an Exercise in Discover/Education OS/2 Database Manager Introduction and Implementation

users perform in the Database Manager course.

The actual tasks that users perform in courses vary, depending on the audience path chosen. For example, a basic user in *Discover/Education OS/2 Introduction and Implementation* learns basic hardware and software concepts, how to use OS/2 commands, and how to use the OS/2 filing system. A support person taking the course learns OS/2 concepts, how to change the configuration file, how to design batch files, and how to use some of the diagnostic features of OS/2.

### Users Control the Pace of Training

Because the courses are modular and user-oriented, Discover/Education courses let users control the pace of training. Users can spend as much time with the course as

needed until they master the material.

**Discover/Education is specially designed to help users master material.** Modules, which last approximately 25 to 45 minutes, concentrate on a limited number of main points. When users finish a module, they feel a well-deserved sense of accomplishment without feeling overwhelmed.

**Discover/Education provides a means for users to determine for themselves whether they have met their learning objectives.** Discover/Education clearly tells users what they can expect to learn, and provides a means for measuring success. Each module begins with a list of objectives that users should be able to achieve upon successful completion. Each module ends with a quiz, which helps users deter-

mine whether the objectives have been met.

**Users take Discover/Education at their own pace.** People learn at different rates, and Discover/Education accommodates these differences by providing many ways for users to learn at their own pace:

- Users can take modules in any order they choose, although the administration program recommends an order. If desired, users can review modules they have completed, or skip ahead to a new module.
- Within modules, there are several sections, called *topics* and *sub-topics*. As with modules, users can take topics and subtopics in any order.
- If users feel they already know the information in a module, topic, or subtopic, they can skip the section entirely.

*Discover/Education is convenient to take.*

**Discover/Education provides a fast path for experienced users.** Having to sit through information that's already known is one of the most frustrating experiences in education. It's also unproductive. In Discover/Education, users who feel they already know the information in a module can review its summary or go directly to the quiz to find out whether they can meet the objectives of the module.

When material in a module in *OS/2 Introduction and Implementation* is similar to information that experienced DOS users may already know, a special topic is provided in the module to teach the differences between DOS and OS/2. If the material is identical, users are told this, and are given the option of skipping the module.

*Once into the course, users learn by doing.*

**Discover/Education uses a variety of teaching techniques to maintain users' interest in the course.**

When their interest is maintained, users are more likely to retain the material and use it on the job.

Discover/Education presents information through easy-to-read text, graphics, and simulation of software about which users are learning. At many points during the course, users have opportunities to practice their skills and to assess what they have learned.

Discover/Education has been carefully developed to make users feel at ease in the computer-based training environment. When answering questions, users never receive rude or overly friendly responses. If users have difficulty answering a question, the computer provides hints and will even provide the correct response.

**Discover/Education is convenient to take.** Because users install the course on their own computing systems, they can take the course whenever their schedule permits. Users don't have to wait for the

availability of an instructor. Because the course is modular, users can take as much of the course as time allows and still feel a sense of accomplishment. And if users need to leave in the middle of a module, Discover/Education lets them set a bookmark. When they return to the

course, users return at the point where they left off.

**Sample Session**

Here's how users start a Discover/Education course for the first time. They begin by specifying a

IBM Discover/Education
<b>Specify Your Student ID</b>
Type the requested information; then press Enter.
Student ID . . . MICHAEL_____
Your student ID can have as many as 20 characters. For example, your name could be your student ID. For a list of IDs currently in use, type a question mark (?).
Enter F1= Help F3=Exit

Figure 3. Panel on Which Users Specify Their Student ID

IBM Discover/Education
<b>WELCOME</b>
This is Discover/Education (™).
Below are two choices. You can choose to read "How to Take Discover/Education," or start a course. If you have never taken a Discover/Education course, it is recommended that you choose option 1.
Type an option number; then press Enter.
1. Read "How to Take Discover/Education" 2. Start a course.
Selection 1
Enter F1= Help F3=Exit

Figure 4. Discover/Education Welcome Panel

student ID. Specifying a student ID is like registering in school. Just as a school registrar uses a student ID to maintain student records, so Discover/Education uses a student ID to maintain student records.

Figure 3 shows the Specify Your Student ID panel. In this example,

the user specifies the student ID, MICHAEL.

Next, users see the Welcome panel, which gives them an opportunity to read "How to Take Discover/Education," a new student orientation session.

Figure 4 shows the Welcome panel. In this example, the user chooses to read "How to Take Discover/Education." This section gives users the instructions for taking Discover/Education courses, and also explains the philosophy underlying the courses, much as an instructor might explain his or her philosophy of teaching before beginning to teach.

When users complete "How to Take Discover/Education," they choose a course to take. Several Discover/Education courses can be installed on one system.

Figure 5 shows the panel from which users select courses. Note the options on this panel to Display modules and Display description. These let users display descriptive information about courses that might help them choose one to take. In this example, the user chooses to take *OS/2 Introduction and Implementation*.

After selecting a course, users select an audience path to take through the course. To determine the most appropriate audience path, users can display a list of modules included in an audience path and a description of the audience path.

Figure 6 shows the panel from which users choose an audience path. In this example, the user chooses the Support Person audience path.

Users are now ready to start a course. Users start courses at the Student Options panel. This panel recommends a module to take and identifies which module has a bookmark if a bookmark was set. This panel also presents users with options to take a module (either the next recommended module or

IBM Discover/Education	
<b>Select a Course</b>	
Position cursor at course; type an option; press Enter. 1=Select 5=Display modules 8=Display description	
Option	Course title
1	OS/2 Introduction and Implementation
-	OS/2 Database Manager Introduction and Implementation
Enter F1= Help F3=Exit	

Figure 5. Panel from Which Users Select a Course

IBM Discover/Education	
<b>Select an Audience Path</b>	
Course title . : OS/2 Introduction and Implementation	
Move cursor to desired audience path; type an option; press Enter. 1=Select 5=Display modules 8=Display description	
Option	Audience path title
-	Basic User
-	Advanced User
1	Support Person
-	All Modules in the Course
Enter F1= Help F3=Exit	

Figure 6. Panel from Which Users Select an Audience Path

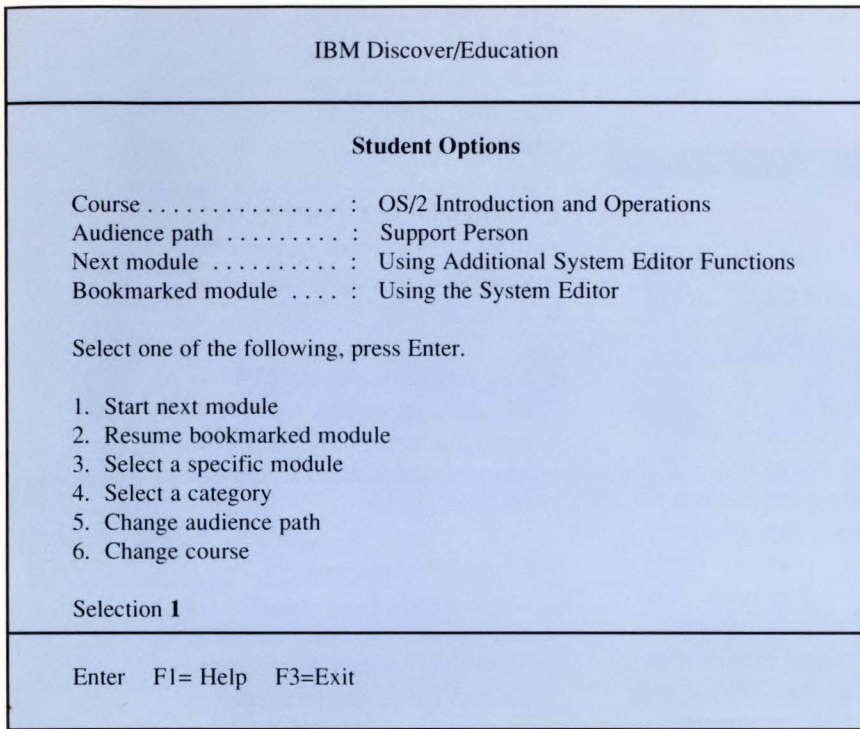


Figure 7. Student Options Panel

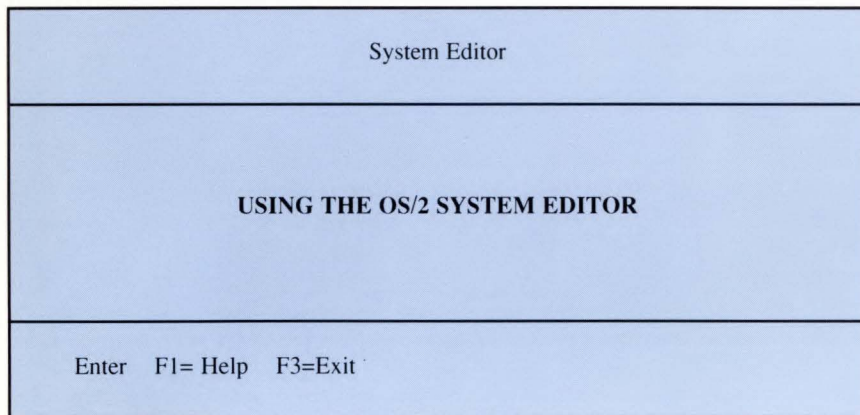


Figure 8. Module Title Panel, the First Panel in a Module

another module), to change audience paths, and to change courses.

Once users are registered in Discover/Education courses, this Student Options panel is the second panel they see. Users do not go through the registration procedure

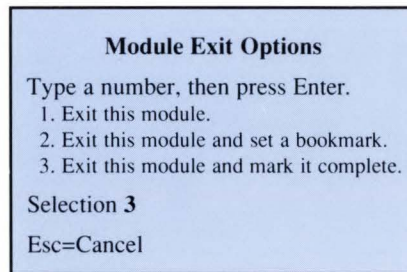


Figure 9. Module Exit Window

each time they start a Discover/Education session.

Figure 7 shows the Student Options panel. In this example, the user has already taken several modules in the course and has set a bookmark in the module, Using the System Editor. The user chooses to take the next module, Using Additional System Editor Functions.

Users next start a module. The first panel in a module is the module title panel, like the one shown in Figure 8.

When users finish a module, or when they need to leave a module, they press F3 to exit the module. The Exit window appears. This window presents many options for leaving the module, including one that lets users return to the module at the point they left and one that updates users' student records.

Figure 9 shows the Module Exit window. In this example, the user chooses to exit the module and mark it complete. This updates the user's student record.

### Handling Additional Training Requirements

Discover/Education can be customized to meet the special needs of your organization. Organizations that use customized applications which require training might consider writing their own modules to meet these unique needs. You can write your own modules using SEF/PC (Self Education Facility for the IBM Personal Computer), the same authoring system used to develop Discover/Education. You can add these modules to one of the Discover/Education courses, or create your own courses. The modules you write can teach users


about those applications or about other office procedures and job skills.

In addition to the OS/2 course, you can purchase Discover/Education courses about IBM DOS, IBM PC and PS/2, IBM AS/400, and IBM System/36.

### **Cost-Effective Training**

Discover/Education can provide cost-effective training. Your OS/2 training may be critical, but the cost of traditional training can be prohibitive. Training budgets often allow for only a few technical staff members to receive training. While this person is in class, users wait for much-needed support. Similarly, end users struggle through the system until they can operate it, often

making countless mistakes, some serious.



### *Discover/Education can provide cost-effective training.*

Discover/Education can resolve these problems. Because it is a computer-based course that can be installed on almost any IBM personal computing system, no travel is required to take the course. The modular nature of the course lets you train many users with the one course. Users can train at their own convenience. And, if someone

leaves your organization, replacement employees can be easily trained at no additional cost (other than time), when and where the training is required.

### **Product Order Numbers**

Product order numbers for the Discover/Education courses discussed in this article are:

- *Discover/Education OS/2 Introduction and Implementation:* order numbers 32F8330 (3.5-inch diskette) and 32F8329 (5.25-inch diskette).
- *Discover/Education OS/2 Database Manager Introduction and Operations:* order numbers 32F8334 (3.5-inch diskette) and 32F8333 (5.25-inch diskette).



## Performance Improvement: A Case Study

Wayne Hammond  
IBM Corporation  
Boca Raton, Florida

In Issue 1, 1988 of this publication we looked at several performance aspects of a microcomputer system. As a follow-up, let's see how (or if) we can utilize benchmarks in our search for improved performance.

### Functional and Performance Benchmarks

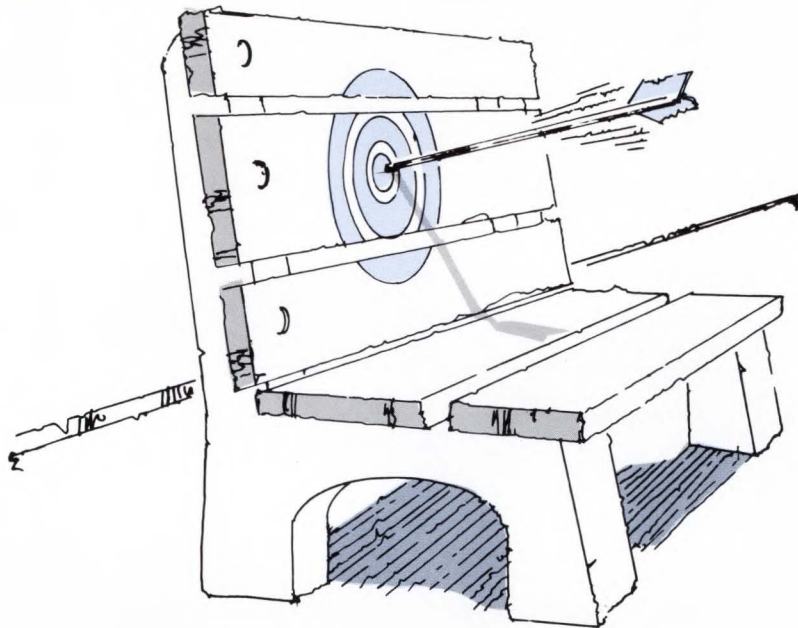
A functional benchmark (that is, a demonstration) is an excellent tool for determining whether a given system is capable of doing a certain type of work. However, it has been my experience that performance benchmarks are often misinterpreted.

When you perform a benchmark, it is essential that you control the environment so that the same workload is executed on the benchmark (test) machine as on your production machine. The workload must be your own workload, or representative of your workload; if it is not, then incorrect conclusions will result.

Let's look at some considerations and difficulties in benchmarking your own work.

### An Example

There was a situation in which Mr. X, a supplier, suggested to Mr. Y, a user, that he get a new



Gee Whiz computer that is clocked much faster than his Old Reliable machine. Mr. Y requested a performance benchmark and chose his most time-consuming application, which had many frequently-used overlays and significant disk input / output (I/O).

### *Performance benchmarks are often misinterpreted.*

Old Reliable had a large, much-used fixed disk that was divided into four logical disk drives. The disk had gone through many allocations and deletions, so that it had many non-contiguous areas for the data files and programs. It also turns out that his multiple overlays were in the first logical drive and his data was in the last logical file; thus, there were frequent long seeks.

When they prepared the benchmark, there was just one job stream, so there was no need for multiple logical disk drives. They loaded everything onto the fixed disk and used only the first few cylinders (short seeks). Further, they cleaned up the disk organization by copying to diskette and then back to a fixed disk, so that all data and programs were contiguous.

The benchmark time was reduced to about 60 percent of what it had been. Mr. Y was convinced of the value of the faster processor, and he purchased several new Gee Whiz computers.

Everything was fine for a while. Then, gradually, the new computers seemed to slow down, and the application requirements continued to grow. Mr. Y needed immediate relief. He decided to use one of the Old Reliable machines that had become available after the completion of a development project.

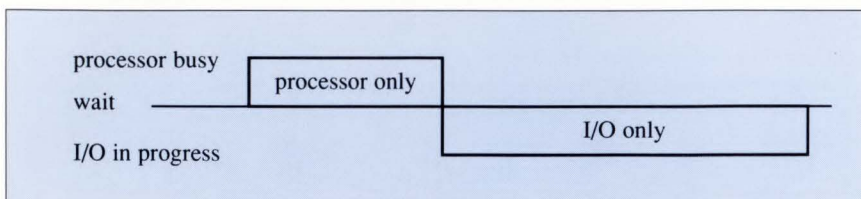


Figure 1. Old Reliable with Poorly Organized Disk

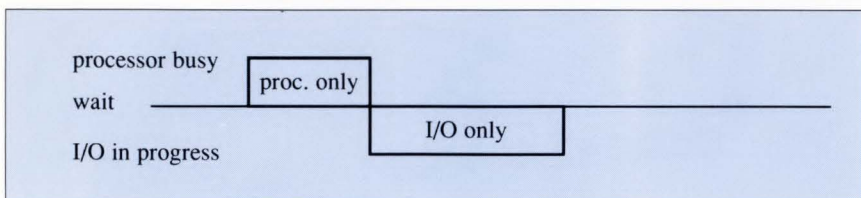


Figure 2. Gee Whiz Benchmark

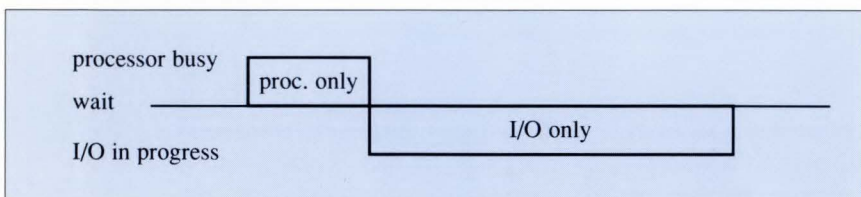


Figure 3. Gee Whiz with Poorly Organized Disk

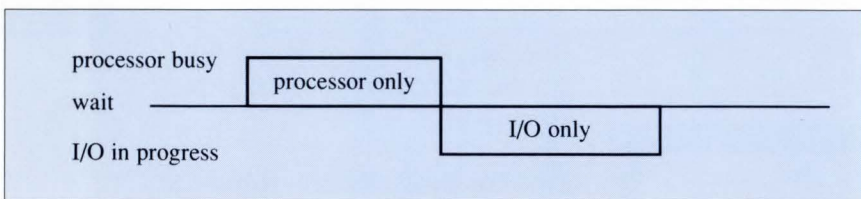


Figure 4. Old Reliable with Organized Disk

Because Old Reliable was now dedicated, its fixed disk drive was reformatted, and the program was loaded into minimal, contiguous disk space. The result was fewer, shorter seeks and better performance. Imagine Mr. Y's surprise upon learning that Old Reliable was outperforming the new Gee Whiz machines, even though the Gee Whiz benchmark was faster.

Mr. Y realized that the faster processor wasn't the only reason that the Gee Whiz benchmark was faster. He continued to study the situation.

He learned how DOS manages disk space and how this can result in non-contiguous areas that cause additional seeks. Mr. Y recognized that, while each seek is fast by human standards, it slows process-

ing. Also, the effects of many seeks are cumulative; that is, as additional seeks are generated, the application continues to run more and more slowly.

Mr. Y concluded that getting the data into the computer system, and getting the data out of the computer, had a greater effect on his program than did the speed of the processor. Mr. Y looked for techniques to improve disk processing, and purchased a utility program that reorganizes the disk and eliminates non-contiguous areas. He runs this program weekly. (Depending on your applications, you might want to run such a program daily, weekly, or monthly.)

Figures 1 through 4 show how Mr. Y's situation looked in state diagrams. In Figure 4, note the approximate balance between processor and I/O. Each plays a significant part in the total elapsed time. If you need to further reduce the time for your application, you can achieve it with either a faster processor or faster I/O. A more complete solution would be to reduce both.

On the other hand, if you just need the ability to do other work, you should look at the lengthy idle times. You can accomplish a lot of extra work just by keeping the machine running.

If a knowledgeable user had this much trouble understanding a benchmark situation using his own application, think about the difficulty in trying to draw correct conclusions from some generalized benchmark results.

## Another Example

Here is another example that shows how easily unexpected results can occur. Mr. Y needed to do another benchmark, this time with application B. He was interested in the elapsed time and repeatability of results. So he ran application B on his production machine to get an accurate time. Then he copied everything to diskette to transfer to the benchmark Gee Whiz machine.

Mr. X loaded the diskettes on the Gee Whiz fixed disk and ran application B. It took longer than Mr. Y's measured run on the production machine. As you probably guessed, Mr. Y was upset with Mr. X, who had told him that the Gee Whiz was a faster machine.

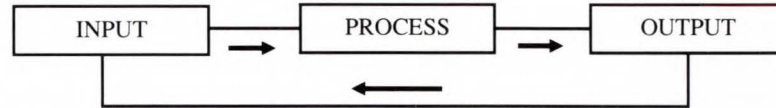
Both Mr. X and Mr. Y knew about poorly organized files, so they reorganized the fixed disk and ran application B again. This time it took several minutes longer. They were both puzzled and thought they might be experiencing a soft error on the disk. A soft error means that a device spends a lot of time in error recovery, but manages to recover so that there isn't any error indication other than long execution times.

Fortunately they persisted, rather than assuming that the Gee Whiz was no good. They used a second Gee Whiz and loaded its fixed disk with Mr. Y's original diskettes. The run time of this second machine matched the first run time of the first Gee Whiz machine.

To understand what happened to our benchmarkers, let's look at a simple definition of a computing system:



and then modify it as follows:



Application B is a bill-of-materials processor. In it, each pass through the system breaks the current level of assembly or subassembly into its components; therefore each run of application B's job stream had additional processing to do.

When they finally did it right by using identical input and options on Mr. Y's production machine and on Mr. X's benchmark machine, they demonstrated that the Gee Whiz machine is indeed much faster than the Old Reliable machine.

Accurate performance benchmarks are possible, but difficult to attain. Question the results until you understand what is happening.

## Standard Benchmarks

Standard benchmarks are intended to allow reviewers to evaluate new systems and tell their readers which hardware performs the best, generally based on price versus performance.

I have doubts that the reviewers are able to accurately compare the throughput capability of each machine, even in the standard benchmark environment. To be able to do so would require running each system in an identical manner, which wouldn't allow them to take advantage of all of a particular

system's features, such as a second fixed disk (which can minimize seeks due to head stealing), or the effect of a math coprocessor (included with the Gee Whiz Plus).

Or, if they did use all of the new features, how much of the performance can be attributed to these features? How would the Old Reliable machine run if equipped with similar features? And so extrapolation (guessing) begins. A guess by an expert is still a guess, but hopefully with a smaller margin of error.

Let's assume that the reviewers are 90 to 100 percent accurate. How much will knowledge about this standard benchmark help you in trying to determine how a particular machine will run your workload? Now you are the person making the guess.

*Read all the footnotes and appendices.*

To help make your guess better, study the benchmark and any information about how it was run; read all the footnotes and appendices. I recently heard of a company that was very impressed by some benchmark performance timings A through G. Unfortunately, they had overlooked a footnote that said if a

particular thing happens – as it did in this case – the program didn't run correctly, so ignore timings A through G.

Count the benchmark elements that are similar to what you do; throw out the parts that you never do; and discount portions that are only slightly similar to what you do. Then consider how much of your total workload is represented by your counted and discounted items, and how closely the DASD used in the benchmark comes to your DASD, and how well organized the DASD is, and how .... Then decide (guess) how well your work will run.

If your analysis of your workload primarily indicates that your processor is busy, a faster processor will help. However, you should look further. If you do a lot of numeric computing, if you are working with real numbers (also known as floating-point numbers) and not integer numbers, and if your programs provide coprocessor support, then you should consider getting a math coprocessor, which offers a tremendous performance gain compared to a faster clock rate on the main processor. If you can take advantage of a math coprocessor, you will achieve increased speed, greater accuracy, and the ability to handle much larger numbers.

If your analysis of your workload indicates a lot of I/O, faster DASD devices will help. Also, to reduce access-arm contention, consider the use of multiple devices, and consider data set (file) placement. To eliminate unnecessary seeks, consider DASD reorganization. And to eliminate mechanical delays, consider using VDISK and/or cache memory.

Finally, don't overlook the idle time – remember that a slow system that is doing something accomplishes more than a fast system that is idle. Scheduling may be as important to performance as a faster processor or faster I/O.

## 3.5" Diskette Formats

*Kevin Maier  
IBM Corporation  
Boca Raton, Florida*

The original recommendations about the proper formatting and use of PS/2 diskettes have undergone revision. This article explains why the recommendations have changed.

### The Original Caution

Personal System/2 shipping cartons include a sheet of paper that cautions users not to format a 2.0 MB diskette to 720 KB, because the diskette then becomes unusable and should be discarded.

This caution was issued because of the physical properties of 720 KB diskettes versus 1.44 MB diskettes. The 720 KB format uses a higher

write current, and the 1.44 MB format uses a lower write current. To accommodate the higher write current, the oxide coating on a 1.0 MB (720 KB formatted) diskette is denser than the oxide coating on a 2.0 MB (1.44 MB formatted) diskette.

When you format a 2.0 MB diskette to 720 KB, you apply the higher write current to the less dense oxide coating. The hardware developers originally felt that this meant the 720 KB formatting pattern is written too deeply into the 2.0 MB oxide coating, causing intermittent data errors and unreliable use. Furthermore, the developers felt that if you attempted to reformat the diskette to 1.44 MB, which uses the lower write current, the 1.44 MB format would not completely write over the "deeper" 720 KB format. Therefore the developers' recommendation was to discard a 2.0 MB diskette that was formatted to 720 KB.

### The Subsequent Findings

Since the time that this caution was issued, the developers have performed additional testing, and have concluded that there is no need to discard a 2.0 MB diskette that was formatted to 720 KB.

It is still true that a 2.0 MB diskette formatted to 720 KB will cause intermittent data errors. However, the latest assessment is that you will be able to reformat the diskette to 1.44 MB and use it reliably after that.

The same logic applies to a 1.0 MB diskette formatted to 1.44 MB. You cannot use it with the 1.44 MB format, but you can reformat it to 720 KB and use it reliably after that.

Therefore, the current recommendation is: If you format a diskette to the wrong capacity, do not discard it; instead, reformat it correctly and use it.

# DASD Details

Wayne Hammond and  
Kevin Maier  
IBM Corporation  
Boca Raton, Florida

The direct-access storage device (DASD) plays a major role in the performance of many computer applications. This article discusses the subject of DASD in depth. It begins with an introduction to DASD, continues with specifications for the DASD devices in the PS/2 family, and concludes with DOS space allocation and I/O timing.

## Introduction

In the simplest view, DASD consists of a rotating magnetic media and a read/write head. Data is mapped onto the media in concentric rings or circles called tracks. The access arm mechanism, or read/write head transport, can be positioned to a particular track, and data can be read or written as the media moves past the read/write head.

There are multiple tracks per media recording surface, and (usually) multiple recording surfaces on the media. For example, a double-sided diskette has two recording surfaces.

*DASD consists of a rotating magnetic media and a read/write head.*

When a device consists of multiple media, the media are arranged so that the tracks on each recording surface are aligned, forming a cylinder. For example, cylinder position 1 includes track 1 of every recording surface. There are as many tracks per cylinder as there are recording surfaces in the device, and there are as many cylinders on the device as there are tracks on a single surface.

The access arm has as many read/write heads as there are tracks in the cylinder. Therefore, when you position the access arm to a particular cylinder (seek), you have access to any or all of the data in the cylinder by selecting (via head

switch) the desired track (read/write head). The data in any cylinder can be accessed without additional movement of the access arm.

The track is further divided into sectors. With DOS, the sector size is 512 bytes. Sectors are written to, or read from, the disk. The number of sectors per track is a function of the device type. See Tables 1 through 11 for the specifications of the devices used with PS/2 systems.

The following steps are required for an I/O operation on a fixed disk:

- (1) The access arm (read/write heads) moves to the proper cylinder (seek);
- (2) The track (head switch) is selected;
- (3) The operation waits for the correct sector to spin under the read/write head (latency);
- (4) The sector is read or written (data transfer).

Typical fixed disk drives rotate continuously at 3600 revolutions per

Table 1. 5.25-Inch External Diskette Drive

Access time:	
Track-to-track .....	6 ms
Head settle time .....	15 ms
Motor start time .....	750 ms
Disk rotational speed:	
300 rpm $\pm$ 1.5%	
200 ms maximum latency	
Formatted characteristics: (double-sided)	
360KB - 4608 data bytes per track	
(9 sectors/track)	
320KB - 4096 data bytes per track	
(8 sectors/track)	
512 data bytes per sector	
40 tracks	
2 data heads	
Sector interleave factor 1:1	
Sector skew factor 0	
Sectors per cluster 2	
Transfer rate: 250,000 bits per second	

Table 2. 3.5-Inch 720KB Diskette Drive

Access time:	
Track-to-track .....	6 ms
Head settle time .....	15 ms
Motor start time .....	500 ms
Disk rotational speed:	
300 rpm $\pm$ 1.5%	
200 ms maximum latency	
Formatted characteristics:	
720KB (360KB per side)	
9 sectors per track	
512 data bytes per sector	
4608 data bytes per track	
80 tracks	
2 data heads	
Sector interleave factor 1:1	
Sector skew factor 0	
Sectors per cluster 2	
Transfer rate: 250,000 bits per second	

Table 3. 3.5-Inch 1.44MB Diskette Drive

Access time:  
 Track-to-track ..... 6 ms  
 Head settle time ..... 15 ms  
 Motor start time ..... 500 ms

Disk rotational speed:  
 300 rpm  $\pm$ 1.5%  
 200 ms maximum latency

Formatted characteristics:  
 1.44M (720K per side)  
 18 sectors per track  
 512 data bytes per sector  
 9216 data bytes per track  
 80 tracks  
 2 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 0  
 Sectors per cluster 1

Transfer rate: 500,000 bits per second

Table 4. 3.5-Inch 20MB Fixed Disk Drive

Access time:  
 Single cylinder seek ..... 15 ms  
 Average seek ..... 80 ms  
 Maximum seek ..... 180 ms

Disk rotational speed:  
 3600 rpm  $\pm$ 0.5%  
 16.67 ms maximum latency

Formatted characteristics:  
 21,000,000 bytes  
 17 sectors per track  
 512 data bytes per sector  
 8704 data bytes per track  
 612 cylinders  
 4 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 2  
 Sectors per cluster 4

Transfer rate: 5.0 Million bits per second  
 Interface: ST412/506 (modified)

Table 5. 3.5-Inch 30MB Fixed Disk Drive

Access time:  
 Single cylinder seek ..... 15 ms  
 Average seek ..... 39 ms  
 Maximum seek ..... 100 ms

Disk rotational speed:  
 3600 rpm  $\pm$ 0.5%  
 16.67 ms maximum latency

Formatted characteristics:  
 30,000,000 bytes  
 25 sectors per track  
 512 data bytes per sector  
 12800 data bytes per track  
 615 cylinders  
 4 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 2  
 Sectors per cluster 4

Transfer rate: 7.5 Million bits per second  
 Interface: ST 506

Table 6. 5.25-Inch 44MB Fixed Disk Drive

Access time:  
 Single cylinder seek ..... 10 ms  
 Average seek ..... 40 ms  
 Maximum seek ..... 80 ms

Disk rotational speed:  
 3600 rpm  $\pm$ 0.5%  
 16.67 ms maximum latency

Formatted characteristics:  
 44,000,000 bytes  
 17 sectors per track  
 512 data bytes per sector  
 8704 data bytes per track  
 733 cylinders  
 7 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 2  
 Sectors per cluster 4

Transfer rate: 5.0 Million bits per second  
 Interface: ST412/506

Table 7. 3.5-Inch 60MB Fixed Disk Drive

Access time:  
 Single cylinder seek ..... 8 ms  
 Average seek ..... 27 ms  
 Maximum seek ..... 45 ms

Disk rotational speed:  
 3600 rpm  $\pm$ 0.5%  
 16.67 ms maximum latency

Formatted characteristics:  
 60,000,000 bytes  
 26 sectors per track  
 512 data bytes per sector  
 13312 data bytes per track  
 762 cylinders  
 6 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 9  
 Sectors per cluster 4  
 Spare sectors per cylinder 6

Transfer rate: 8.4 Million bits per second  
 Interface: ESDI (modified)

Table 8. 5.25-Inch 70MB Fixed Disk Drive

Access time:  
 Single cylinder seek ..... 5 ms  
 Average seek ..... 30 ms  
 Maximum seek ..... 60 ms

Disk rotational speed:  
 3600 rpm  $\pm$ 0.5%  
 16.67 ms maximum latency

Formatted characteristics:  
 73,000,000 bytes  
 36 sectors per track  
 512 data bytes per sector  
 18432 data bytes per track  
 583 cylinders  
 7 data heads  
 Sector interleave factor 1:1  
 Sector skew factor 1  
 Sectors per cluster 4  
 Spare sectors per cylinder 4

Transfer rate: 10 Million bits per second  
 Interface: Enhanced Small Device Interface

Table 9. 5.25-Inch 115MB Fixed Disk Drive

Access time:	
Single cylinder seek .....	6 ms
Average seek .....	28 ms
Maximum seek .....	60 ms
Disk rotational speed:	
3600 rpm $\pm 0.5\%$	
16.67 ms maximum latency	
Formatted characteristics:	
115,000,000 bytes	
36 sectors per track	
512 data bytes per sector	
18432 data bytes per track	
915 cylinders	
7 data heads	
Sector interleave factor 1:1	
Sector skew factor 11	
Sectors per cluster 4	
Spare sectors per cylinder 4	
Transfer rate: 10 Million bits per second	
Interface: Enhanced Small Device Interface	

Table 10. 3.5-Inch 120MB Fixed Disk Drive

Access time:	
Single cylinder seek .....	8 ms
Average seek .....	23 ms
Maximum seek .....	40 ms
Disk rotational speed:	
3600 rpm $\pm 0.5\%$	
16.67 ms maximum latency	
Formatted characteristics:	
120,000,000 bytes	
32 sectors per track	
512 data bytes per sector	
16384 data bytes per track	
925 cylinders	
8 data heads	
Sector interleave factor 1:1	
Sector skew factor 12	
Sectors per cluster 4	
Spare sectors per cylinder 8	
Transfer rate: 10.2 Million bits per second	
Interface: ESDI (modified)	

Table 11. 5.25-Inch 314MB Fixed Disk Drive

Access time:		Formatted characteristics:	
Single cylinder seek .....	5.5 ms	314,000,000 bytes	
Average seek .....	23 ms	34 sectors per track	
Maximum seek .....	50 ms	512 data bytes per sector	
Disk rotational speed:		17408 data bytes per track	
3283 rpm $\pm 0.5\%$		1225 cylinders	
18.28 ms maximum latency		15 data heads	
Transfer rate: 10 Mbps		Sector interleave factor 1:1	
Interface: Enhanced Small Device Interface		Sector skew factor 8	
		Sectors per cluster 4	

minute. Diskette drives rotate at 300 rpm, but only on demand; that is, the drive motor is turned on and the device must come up to speed before I/O can be done, and, after the I/O is complete, it is turned off and coasts to a stop. This adds a significant amount of time to diskette I/O operations.

## PS/2 DASD Specifications

Tables 1 through 11 give performance specifications for DASD devices used in the PS/2 family. In these tables, all access times are in milliseconds (ms), and average seek time is defined as the time required for the access arm to traverse one-third of the cylinders. Fixed disk

head settle time is included in the access time.

Fixed disk seek time is nonlinear — a major portion of single-cylinder seek time is spent just getting the head moving (overcoming inertia), followed by a fairly linear cylinder-to-cylinder time that is additive — the farther you seek, the longer it takes.

Diskette seek time is linear because of the use of a stepper motor to move the read/write heads. The time can be computed as [the number of tracks moved times the track-to-track seek time] plus the head-settle time.

The latency times shown are for a full rotation (maximum). Latency varies from zero to maximum, I/O timing calculations usually use one-half of maximum latency.

### *Fixed disk seek time is nonlinear.*

To time diskette I/O operations, you must allow for motor-start time, seek-time, head-settle time, latency, and data-transfer time.

Fixed disk timings are similar, but they don't require motor-start times,

and head-settle time is included in the seek times given.

Sector skew factor is the number of sectors that the start of a given track is displaced from the start of its predecessor track. Sector skewing is used to compensate for head-switching time when multiple-sector read operations cross a head boundary; otherwise, the next sector would be missed and the device would have to wait for a full revolution of the disk.

A basic definition of sector interleave factor is the number of disk rotations required to read a track of information. For example, a factor of 2:1 would require 2 rotations of the disk for each track.

## DOS Format and Space Allocation

When DOS is used to format a diskette, the boot record is written on track 0, sector 1, side 0. The boot record is used to produce an error message if you try to start up the system with a nonsystem diskette in drive A. For fixed disks, the boot record resides in the first logical sector of a DOS partition.

Following the boot record are two copies of the FAT (File Allocation Table). The FAT is used by DOS to manage DASD space allocation, and has an entry for each cluster on the device. The cluster is the unit of space allocation (or deallocation), and is either 1, 2, or 4 sectors per cluster for PS/2 devices (see Tables 1 through 11).

The size of the FAT varies by device, depending on the amount of space to be mapped, the number of bits per FAT entry, the number of sectors per cluster, and so on. Ex-

amples in this article have 16-bit entries in the FAT. 12-bit FAT entries are used by diskette drives and fixed disk partitions up to 10 MB.

The root directory follows the FAT tables. Space beyond the root directory is used for objects (data files, programs, and subdirectories). Each directory entry has a FAT pointer, which is the first cluster of that object.

The FAT entries, for each object, are constructed like a linked list. That is, the first cluster points to (has the address of) the second cluster, which points to the third cluster, and so on. The list ends when the Nth cluster is marked as the end of the list (FFFFH). Single-cluster objects begin and end with the same cluster (one FAT entry).

Unallocated clusters are marked (in the FAT) as 0000H. When an object is erased (deleted), its FAT entries are reset to 0000H, and the space the object had occupied is now available to be allocated. FFF7H is used to mark a cluster as defective.

DOS manages a pointer to the free space on the DASD, and allocates space as required. This space allocation could be for new objects, or it could be for objects being ex-

tended. Therefore, normal allocation and deletion (especially when combined with extension of some objects) will result in objects that do not have contiguous space allocated.

This is normal, and the system continues to run; however, you may experience degraded performance. If you are doing sequential processing of data files or loading programs, these non-contiguous areas will cause additional seeks and latency delays. The impact of this performance degradation can vary from barely noticeable to severe – the amount of degradation depends on your situation. You probably won't notice much difference if you are processing the non-contiguous object in a random manner.

The DOS utility CHKDSK can be used to check the condition of your device; for example, CHKDSK d:\path\\*.\* provides information about the number of non-contiguous areas in the specified subdirectory. It is probable that there are commercial programs that "walk" through the entire directory structure and report on the condition of the entire device.

If you determine that non-contiguous areas are degrading performance, you should reorganize your device to eliminate the problem. You can use a couple of

ATTRIB	+A *.*	(turn on archive bits)
XCOPY	*.* A: /M	(copy to diskette, turns archive bit off - ends if diskette full) (repeat until all objects copied)
DEL	*.*	(erase the subdirectory)
XCOPY	A:\	(reload objects) (repeat for each diskette copied)

Figure 1. A DOS Technique to Reorganize a Subdirectory



<b>1.44 MB diskette best case:</b>			
Average seek	= (80 / 3 * 6) + 15	=	175 ms (see Note 1)
Average latency	= 200 / 2	=	100 ms
Transfer time	= 200 * 1/9	=	<u>22.2</u> ms (see Note 2)
			297.2 ms/record
Total time	= 297.2*2800+500	=	832660 ms
		=	832.66 seconds
		=	13.88 minutes
<b>1.44 MB diskette worst case:</b>			
Motor start time (see Note 3)	=	500	ms
Average latency	= 200 / 2	=	100 ms
Transfer time	= 200 * 1/9	=	<u>22.2</u> ms
			622.2 ms/record
Total time	= 622.2 * 2800	=	1,742,160 ms
		=	1742.16 seconds
		=	29.04 minutes
<b>Notes:</b>			
1.	Average seek for diskette drive = 1/3 the number of tracks times access-time-per-track plus head-settle time.		
2.	Transfer time is latency time multiplied by the number of sectors transferred divided by sectors per track.		
3.	No seek time is shown, because the seek completes during motor start time.		

Figure 2. 1.44 MB Diskette Drive Timing Random I/O

<b>70 MB fixed disk:</b>			
Average seek	= 30	=	30 ms
Average latency	= 16.67 / 2	=	8.34 ms
Transfer time	= 16.67 * 1/36	=	<u>0.46</u> ms
			38.80 ms/record
Total time	= 38.80 * 2800	=	108,640 ms
		=	108.64 seconds
		=	1.81 minutes

Figure 3. 70 MB Fixed Disk Drive Timing Random I/O

1)	Data bytes/track	=	sectors/track * data bytes/sector
2)	Data bytes/cylinder	=	data bytes/track * number of data heads
3)	Latency (ms)	=	1000 * 60 / rotational speed (in rpm)
4)	Time per sector (ms)	=	latency / (sectors/track)
5)	Time per cluster (ms)	=	time/sector * sectors/cluster
6)	Single-track transfer rate (bytes per second) =		
			(data bytes/track * 1000 / latency) / sector interleave factor
7)	Single-cylinder transfer rate (bytes per second) =		
			data bytes/cylinder * 1000 / ((latency * number of data heads) + (number of data heads - 1) * sector skew * time/sector)

Figure 4. Sequential I/O Timing Formulas

approaches. The simplest technique is to use a "utility program" that reorganizes all of the objects on the device. Another, more cumbersome, approach uses the facilities provided by DOS to partially reorganize the default subdirectory. This technique may not yield a perfect reorganization, but it is effective when the subdirectory has many non-contiguous blocks (see the example in Figure 1).

## DASD I/O Timing

Whenever DASD I/O is part of a process, the minimum time for that process is the I/O time. No matter how fast your processor is, or how much processor-I/O overlap you achieve, the minimum process time is the I/O time. The actual process time will be longer because there is always some overlapped processing.

If you need to reduce this I/O time, you must either use faster devices or find a way to reduce the number of actual I/O operations.

Let's look at some timing examples. First, random I/O with the following requirements:

- 512-byte records (1 sector)
- 2800 records to be processed
- Direct (random) processing with average seeks

Figure 2 is for the 1.44 MB diskette drive. In the best-case example, the process is fast enough that we only have to wait for the initial diskette motor start time. In the worst-case example, the process is slow enough that we have to wait for diskette motor start on each record.

<b>70 MB Fixed Disk</b>	
single-track rate	= $18432 * 1000 / 16.67 = 1,105,699$ bytes per second
single-cylinder rate	= $18432 * 7 * 1000 / ((16.67 * 7) + (6 * 1 * 16.67/36)) = 1,079,985$ bytes per second
<b>1.44 MB Diskette Drive</b>	
single-track transfer rate	= single-cylinder transfer rate = $9216 * 1000 / 200 = 46,080$ bytes per second

Figure 5. Maximum Transfer Rates – Sequential I/O

<b>1.44 MB Diskette Drive</b>	
motor start	= 500 ms
assume an average seek to start of diskette	= 0 ms (Note 1)
track-to-track seeks = $79 * (6+15)$	= 1659 ms
latency per seek = $(200 - 21) * 79$	= 14141 ms
transfer time	
80 tracks * 2 heads * 200 ms	= $\frac{32000}{48300}$ ms
	48.3 secs
<b>70 MB Fixed Disk Drive</b>	
data bytes / cylinder = $512 * 36 * 7$	= 129024
# cylinders = $1,440,000 / 129024$	= 11.16
assume average seek to start of data	= 30 ms
cylinder-to-cylinder seeks = $11 * 5$	= 55 ms
assume average latency = $11 * 8.34$	= 91.74 ms
data transfer time/cylinder = $16.67 * 7$	= 116.69 ms
11 full cylinders = $116.9 * 11$	= 1285.9 ms
partial cylinder = $116.9 * .16$	= 18.7 ms
	$\frac{1598.03}{1.6}$ ms
	1.6 secs
<b>Note 1.</b> Zero seek time because seek completes during motor start time.	

Figure 6. Sequential I/O Timing Examples

Figure 3 shows direct (random) I/O activity using the 70 MB fixed disk. As expected, the fixed disk is clearly superior to a diskette.

Figure 4 lists some sequential timing formulas. In Figure 4, the sixth formula (single-track transfer rate) assumes an organized disk (no seeks), and is the fastest theoretical

transfer rate based on the geometry of the device. PS/2 devices have a sector interleave factor of 1:1. Note the impact of the sector interleave factor; for example, a sector interleave of 2:1 would halve the transfer rate.

Formula 7 (single-cylinder transfer rate) has the same assumptions as

formula 6. Formula 7 accounts for sector skew, but not for a cylinder seek. You only have the sector skew factors if you have an interleave of 1:1. Replace everything to the right of the + with the sector interleave factor if you are timing non-PS/2 disks.

Figure 5 shows how to time sequential I/O operations on the 1.44 MB diskette drive and the 70 MB fixed disk drive.

Figure 5 (using formulas 6 and 7) shows the single track transfer rate of the 70 MB fixed disk drive to be 1,105,699 bytes per second and the single cylinder transfer rate to be 1,079,985. (Note that the single-cylinder transfer rate is lower than the single-track transfer rate, because of sector skew.)

For the 1.44 MB diskette drive, the single-track transfer rate equals the single-cylinder transfer rate, and is 46,080 bytes per second.

*As expected, the fixed disk is clearly superior to a diskette.*

Sequential I/O timing is a fast, convenient way to determine the minimum process time based on I/O time. (It assumes zero process time, and shows the capability of the selected DASD). You must add process time in order to estimate the overall elapsed time.

Suppose you are considering a process that must examine a 1.44 MB sequential file and the process must complete in 30 seconds or less.

Figure 6 shows the calculation of the sequential I/O processing times for both devices, and indicates that you must use the 70 MB fixed disk drive with a transfer time of 1.6 seconds. The transfer time for the 1.44 MB diskette drive is 48.3 seconds.

The 70 MB fixed disk example ignores the four spare sectors per cylinder, so it is not 100 percent accurate, but gives good indicative information.

These should be enough examples to give you an idea of the steps involved in I/O timings. Again, these

timings assume the maximum capability of the device, and they only show the I/O time required, which is the least time possible, even with a very high-speed processor.

### **PACKCOPY Program Available on IBM Bulletin Board**

The program in the following article, PACKCOPY, is being made available on the IBM PC User Group Support Bulletin Board.

The bulletin board contains a main message board, seven public conferences about various topics, numerous files to download (bulletins, announcements, software), and three searchable data bases: OS/2-compatible application software, Micro Channel adapter cards, and a PC user group locator service.

The bulletin board runs using PCBoard software. The recommended modem setting is eight bits, no parity (for downloading, this setting is required). The bulletin board can accommodate modems with speeds up to 9600 baud.

The phone number for the IBM PC User Group Support Bulletin Board is 404-988-2790. PC user group officers have access to an 800 number.

# Making a Mirror Image of a Fixed Disk

Kevin Maier  
IBM Corporation  
Boca Raton, Florida

PACKCOPY is an assembly-language utility program that copies a mirror image of the contents of an Enhanced Small-Device Interface (ESDI) fixed disk onto another ESDI fixed disk of the same size.

The purpose of PACKCOPY is to create the same fixed disk configuration on multiple Personal System/2 systems. Without PACKCOPY it could take you from ten minutes to several hours to configure the contents of a single fixed disk. If you have to do the same configuration many times for multiple disks, you could spend several hours or days configuring the fixed disks on all the systems.

If you could perform the equivalent of a DISKCOPY from one fixed disk drive to another, the job of creating multiple identical disks would be much easier. Normally you cannot DISKCOPY one fixed disk to another because of the architecture of most fixed disk drives. PACKCOPY was written to provide the equivalent function of DISKCOPY for ESDI disk drives.

## Using PACKCOPY

Using PACKCOPY is easy. You can enter just the word:

```
PACKCOPY
```

to see a set of instructions, or use the full syntax:

```
PACKCOPY [source] [target]
```


The only two valid entries are:

```
PACKCOPY 0 1
```

or

```
PACKCOPY 1 0
```

where 0 is the first disk drive and 1 is the second disk drive. Because you can specify a drive as either the source or the target, you can copy from drive 0 to drive 1 or vice-versa. PACKCOPY tests to make sure that there are two fixed disks installed, and that they are ESDI drives of the same size. PACKCOPY also tests the two parameters to make sure that one is 0 and the other is 1.



*PACKCOPY creates the same fixed disk configurations on multiple systems.*

You might ask why the syntax of the PACKCOPY command uses numbers (0 and 1) rather than drive letters (C:, D:). The reason is that a large-capacity fixed disk can contain several 32 MB (or smaller) logical drives. For example, a 70 MB disk can contain 32 MB logical drives C and D and a 6 MB logical drive E. Because of this, PACKCOPY cannot work with logical drives. Therefore, the syntax uses physical drives 0 and 1.

Where are physical drives 0 and 1? Take the side cover off a Personal System/2 Model 60 or 80 386, and look inside. The fixed disk drive that comes with a PS/2 Model 60 or 80 386 is positioned toward the rear of the system unit, and is connected to the disk drive adapter via a cable. This is drive 0. Directly across from drive 0, toward the front of the system unit, is room for an optional second fixed disk drive. When in-

stalled, this is drive 1. You install drive 1 by removing the front cover, sliding the drive into position, and then connecting its cables to the disk drive adapter and the power supply. **Important:** Be sure to connect the ground wire.

## Versatility

PACKCOPY can accommodate any situation you may have. You can choose to use PACKCOPY totally within a single PS/2 Model 60 or 80 386, or you can transport PACKCOPY to all the PS/2 Models 60 and 80 386 systems that you have.

If you use PACKCOPY totally within one system (here called System A), you should copy from drive 0 to drive 1. After you copy to drive 1, you should remove drive 1 from System A and install it in System B as either drive 0 or drive 1. Next, into the drive 1 position in System A, insert another target disk drive, and use PACKCOPY again. Then remove drive 1 from System A and install it in System C as either drive 0 or drive 1. Repeat this procedure until you have created all the mirror-image fixed disks that you need.

Your other option is to transport PACKCOPY to all of the PS/2 Models 60 and 80 386 systems that you have. In this case, in System A, make your first copy from drive 0 to drive 1. Then remove drive 1 from System A and install it as drive 1 in System B. Then, on System B, use PACKCOPY to copy from drive 1 to drive 0. This creates a mirror image on drive 0 of System B. Next, remove drive 1 from System B, install it as drive 1 in System C, and, on System C, use PACKCOPY to copy from drive 1 to drive 0. Repeat this procedure

until you have created all the mirror-image fixed disks that you need.

## Operation

The PACKCOPY program has not been optimized for memory usage or for data transfer speeds. The program uses a single 32 KB buffer for transferring data between drives.

The copying process is done at a BIOS level using cylinder/head/sector parameters. As such, it is not dependent on the file management that is done by operating systems. For example, if your source fixed disk is partitioned into three or more logical drives under DOS Version 3.30, all partitions and data will be transferred to the target disk. If your drive is partitioned using multiple operating systems, all data from all partitions will be transferred. **Note:** If the source and target disk drives have different partition definitions before the copy, then after the copy you will need to reboot the system to properly access the target drive and the data it contains.

PACKCOPY requires that both the source and target disk drives be ESDI drives of the same size. In PS/2 Models 60 and 80 386, fixed disk drives of capacities 70 MB and larger are ESDI drives.

PACKCOPY supports only ESDI fixed disks because the formatting of ESDI disk drives allows for zero defects. In contrast, ST-506 fixed disks have defect mapping, which prevents a mirror-image copy program from working. ST-506 fixed disk drives use cylinder, head and sector addressing to access data on the disk. During the manufacturing process, defective areas on the disk are marked so they are not used later for data storage. After

the disk drive is installed in the system and you issue the FORMAT command, PC DOS enters the addresses of the defective sectors into the File Allocation Table (FAT). Obviously you cannot be assured of zero defects, or of identical defective sectors between two fixed disks.

### *The copying process is done at a BIOS level.*

Another problem is the physical layout of the drive – the number of cylinders, heads and sectors. It is possible to have two 44 MB ST-506 disk drives with different numbers of cylinders, heads and sectors. For example, two different 44 MB fixed disks are supplied for the Models 60 and 80 386. The type 31 drive has 732 cylinders, 7 read/write heads, and 17 sectors. The type 32 drive has 1023 cylinders, 5 heads, and 17 sectors. Therefore, even if both drives were defect-free, a mirror copy from one to the other would be impossible.

The ESDI interface solves the ST-506 problems and provides additional benefits as well. The obvious advantage of ESDI drives is faster data transfer rates. A less obvious advantage is the physical architecture. Although the disk still has physical cylinders, heads and sectors, they are not accessed that way. Instead, ESDI uses a Relative Block Address (RBA) value to access the sectors.

Each sector, or block, has an associated RBA value. Because DOS uses cylinder/head/sector (CHS) values for accessing data, a conversion is necessary. The BIOS ROM

contained on the ESDI controller card performs this conversion from RBA to CHS. It also uses a standard set of parameters for head and sector counts. All ESDI drives on PS/2 systems have 64 read/write heads and 32 sectors per track. This translates to 1 megabyte per cylinder (64 times 32 times 512 bytes).

The other advantage of the ESDI drives is defect mapping. During the hardware format, RBAs are assigned to the sectors. If a defective sector is found, an alternate is assigned to it. The result is a fixed disk of the specified capacity with zero defects.

These differences between ESDI and ST-506 make possible a mirror-image copy from one ESDI drive to another.

PACKCOPY also runs under DOS and uses the CBIOS interface for fixed disk (INT 13H). As such, it accesses the disk using CHS addressing.

## Performance

I have tested the PACKCOPY utility on two PS/2 systems – a PS/2 Model 60-071 with 70 MB fixed disk drives, and a PS/2 Model 80-111 with 115 MB fixed disk drives. The 70 MB fixed disk was copied in less than four minutes, and the 115 MB disk was copied in about seven minutes. To these numbers should be added the time it takes to remove the target disk drive after the copying is done, and to insert another target disk drive.

*Editor's note: Users of the PACKCOPY program are cautioned to adhere to the terms and conditions of software license agreements.*

```

*****
;PACKCOPY lets you copy the entire contents from one ESDI-interfaced
;disk drive to another drive of the same type. It simply copies all
;data from one to the next, allowing a mirror-image to be created.
;It is not OS-dependent, and will copy all accessible data.
;
;To use: PACKCOPY 0 1
;
; or
; PACKCOPY 1 0
*****
                CODE  SEGMENT PUBLIC 'CODE'
                ASSUME CS:CODE,DS:CODE

BEGIN          DB      128 DUP(0)      ;PSP header

PARAM         DB      128 DUP(0)      ;Parm string

                ORG    100H            ;Start of program code

START:        JMP     INITIAL         ;Goto start of code

MSG           MACRO  TEXT              ;Print message macro
                PUSH  DX              ;Save register
                MOV   DX,OFFSET TEXT  ;Get address of text
                CALL  SHOW_MSG        ;Call subroutine
                POP   DX              ;Restore register
                ENDM

                DB      ' Written by KE Maier '

;All the variables and pointers used are below

HEAD          DB      0                ;Head ID
CYL_SCT       DB      1                ;Cylinder/sector count
CYL_LOW       DB      0                ;Cylinder count (low 8 bits)

HEADS_I       DB      0                ;New pointer for head
SECTOR_I      DB      1                ;New pointer for sector
CYLINDER_I    DW      0                ;New pointer for cylinder

CYLINDER      DW      0                ;Total cylinders max
SECTOR        DB      0                ;Total sectors max
HEADS         DB      0                ;Total heads max

BLOCK_SIZE    DB      5 DUP(0)        ;Number of 32KB blocks to
;process (ASCII)

BLOCKS        DW      0                ;Block count to transfer
COUNT        DW      0                ;No. 32kb blocks transferred

SOURCE_D      DB      0                ;Source drive
TARGET_D      DB      0                ;Target drive
HF_COUNT      DB      0                ;Fixed disk count

HEAD_0        DB      0                ;First drive parms
CYL_SCT_0     DB      0
CYL_LOW_0     DB      0

HEAD_1        DB      0                ;Second drive parms
CYL_SCT_1     DB      0
CYL_LOW_1     DB      0

DIVISOR       DB      64               ;No. of sectors/32KB block
TEMP0         DB      0                ;Temps for divide routines
TEMP1         DB      0

BUFFER        DB      4096 DUP(' BUFFER ')
;Reserve 32KB buffer

```

Figure 1. PACKCOPY Assembler Source Code (Part 1 of 10)

```

INITIAL:
                CALL  GET_PARM        ;Parm string from PSP header
                JNC   SKIP_INST      ;Parms are present,
;try to parse and run
                MSG   INSTRUCTIONS   ;Print instructions
                INT   20H            ;Exit to DOS

SKIP_INST:
                MSG   INTROMSG       ;Print intro message
                CALL  PARSE_LINE     ;Parse the parm line for usage
                JNC   S_INST         ;Continue if no errors
                MSG   B_PARMS        ;Print bad parameter message
                INT   20H            ;Exit to DOS

S_INST:
                MOV   AL,SOURCE_D    ;Get source fixed disk
                CMP   AL,TARGET_D    ;Compare for same as target
                JNE   SKIP_0         ;If different, continue
                MSG   CONFLICT       ;Drive conflict exists
                INT   20H            ;Exit to DOS

SKIP_0:
                MOV   AL,SOURCE_D    ;Get source fixed disk
                CMP   AL,80H         ;Is it drive zero?
                JE    DO_TARGET      ;Yes, verify target
                CMP   AL,81H         ;Is it in range?
                JE    DO_TARGET      ;Good, now verify target
                MSG   OUT_RANGE      ;Out-of-range message
                INT   20H            ;Exit to DOS

DO_TARGET:
                MOV   AL,TARGET_D    ;Get target drive
                CMP   AL,80H         ;Is It drive zero?
                JE    DRV_OK         ;Drives okay, go for it!
                CMP   AL,81H         ;Is it drive one?
                JE    DRV_OK         ;Drives okay, go for it!
                MSG   OUT_RANGE      ;Out-of-range message
                INT   20H            ;Exit to DOS

DRV_OK:
                CALL  TEST_ESDI      ;Make sure drives are
;ESDI-interfaced
                JNC   SKIP_1         ;If okay, continue
                MSG   NOT_ESDI       ;Not ESDI drives, exit
                INT   20H            ;Exit to DOS

SKIP_1:
                MOV   AL,HF_COUNT     ;Get drive count
                CMP   AL,2           ;Two drives?
                JE    SKIP_2         ;Yes, continue
                MSG   BAD_COUNT      ;Print message
                INT   20H            ;Exit to DOS

SKIP_2:
                CALL  DISK_PARMS     ;Get disk parms
                CALL  DISK_SIZE      ;Check for same drive size
                JNC   SKIP_3         ;If okay, continue
                MSG   BAD_DRIVES     ;Drives different
                INT   20H            ;Exit to DOS

SKIP_3:
                MSG   DRV_OKAY       ;Print drive okay message
                CALL  CALC_BLKs      ;Calculate total disk blocks
;in 32KB
                CALL  PRINT_BLOCK    ;Print number of blocks
                MSG   WARNING        ;Print warning message

CONT_LOOP:
                MSG   CONTINUE       ;Print continue message
                MOV   AH,01H         ;Get keyboard input w/echo
                INT   21H            ;Call DOS
                CALL  MAKE_LC        ;Convert to lower case
                CMP   AL,'y'         ;Yes?
                JE    SKIP_4         ;Continue
                CMP   AL,'n'         ;No?
                JE    DONTDOIT       ;Abort program

PROMPT_LOOP:
                MSG   WRONG_KEY      ;Print message
                JMP   CONT_LOOP      ;Loop back

```

Figure 1. PACKCOPY Assembler Source Code (Part 2 of 10)

```

DONTDOIT:
    MSG ABORT ;Issue abort message
    INT 20H ;Exit to DOS

SKIP_4:
    MSG RUSURE ;Make sure user wants to
    MSG CONTINUE ;Prompt user again
    MOV AH,01H ;Get keyboard input w/echo
    INT 21H ;Call DOS
    CALL MAKE_LC ;Convert to lower case
    CMP AL,'y' ;Yes?
    JE SKIP_5 ;Continue
    CMP AL,'n' ;No?
    JE DONTDOIT ;Abort program
    JMP PROMPT_LOOP ;Loop back

SKIP_5:
    MSG PROCEED ;Proceed message
    MOV CX,BLOCKS ;No. of 32KB blocks for size

XFER_LOOP:
    CALL READ_BLOCK ;Read a block (32KB)
    CALL WRITE_BLOCK ;Write a block (32KB)
    INC COUNT ;Increase block count
    CALL PRINT_COUNT ;Print no. 32KB blocks read
    CALL UPDATE_POINT ;Update pointers
    CALL XFER_POINT ;Transfer new pointers
    LOOP XFER_LOOP ;Loop back until done

    MSG DATA_XFER_CMP;Data transfer complete msg.
    INT 20H ;Exit to DOS

READ_BLOCK PROC NEAR
    PUSH AX ;Save registers
    PUSH BX
    PUSH CX
    PUSH DX

    MOV AH,02H ;Get read command
    MOV DL,SOURCE_D ;Get drive ID
    MOV DH,HEAD ;Get head number
    MOV CL,CYL_SCT ;Get upper cylinder/sector ID
    MOV CH,CYL_LOW ;Get lower cylinder count

    LEA BX,BUFFER ;Point to buffer

    MOV AL,64 ;Read 32KB (64 * 512)
    INT 13H ;Call BIOS
    JC R_ERROR ;Exit if error

    POP DX ;Restore registers
    POP CX
    POP BX
    POP AX

    RET ;Return to caller
READ_BLOCK ENDP

R_ERROR:
    MSG R_ERROR1 ;Print error
    JMP ERROR_EXIT ;Clear stack and exit

WRITE_BLOCK PROC
    PUSH AX ;Save registers
    PUSH BX
    PUSH CX
    PUSH DX

    MOV AH,03H ;Get write command

```

```

    MOV DL,TARGET_D ;Get drive ID
    MOV DH,HEAD ;Get head number
    MOV CL,CYL_SCT ;Get upper cylinder/sector ID
    MOV CH,CYL_LOW ;Get lower cylinder count

    LEA BX,BUFFER ;Point to buffer
    MOV AL,64 ;Read 32KB (64 * 512)

    INT 13H ;Call BIOS
    JC W_ERROR ;Exit if error

    POP DX ;Restore registers
    POP CX
    POP BX
    POP AX

    RET ;Return to caller
WRITE_BLOCK ENDP

W_ERROR:
    MSG W_ERROR1 ;Print error

ERROR_EXIT:
    POP DX ;Clear stack
    POP CX
    POP BX
    POP AX
    MSG TERMINATE ;Print terminate message
    INT 20H ;Exit to DOS

PRINT_COUNT PROC
    PUSH DI ;Save registers
    PUSH DX
    PUSH CX
    PUSH AX

    MOV DX,COUNT ;Get transfer count
    MOV DI,0006D ;Get offset into string

    MOV CX,0000H ;Initialize counter
    LEA DI,ASCII_COUNT_1;Point to buffer location

DEC16LOOP1:
    PUSH CX ;Save count
    MOV AX,DX ;Numerator
    MOV DX,0 ;Zero upper half
    MOV CX,10 ;Divisor of 10
    DIV CX ;Divide
    XCHG AX,DX ;Get quotient

    ADD AL,30H ;Make ASCII decimal
    MOV [DI],AL ;Put into string
    DEC DI ;Decrement to next byte

    POP CX ;Restore CX for count
    INC CX ;Increment to next digit
    CMP DX,0 ;Done yet?
    JNZ DEC16LOOP1 ;No, loop back

    MSG ASCII_COUNT ;Print string

    POP AX ;Restore registers
    POP CX
    POP DX
    POP DI

    RET ;Return to caller
PRINT_COUNT ENDP

PRINT_BLOCK PROC
    PUSH DI ;Save registers
    PUSH DX
    PUSH CX

```

Figure 1. PACKCOPY Assembler Source Code (Part 3 of 10)

Figure 1. PACKCOPY Assembler Source Code (Part 4 of 10)

```

        PUSH    AX

        MOV     DX,BLOCKS    ;Get XFER count
        MOV     DI,0006D    ;Get offset into string
        MOV     CX,0000H    ;Initialize counter
        LEA    DI,ASCII_BLOCK_1;Point to string location

DEC16LOOP2:
        PUSH    CX          ;Save count
        MOV     AX,DX       ;Numerator
        MOV     DX,0       ;Zero upper half
        MOV     CX,10      ;Divisor of 10
        DIV    CX          ;Divide
        XCHG   AX,DX       ;Get quotient

        ADD    AL,30H      ;Make ASCII decimal
        MOV     [DI],AL    ;Put into string
        DEC    DI          ;Decrement to next byte

        POP    CX          ;Restore CX for count
        INC    CX          ;Increment the digit
        CMP    DX,0        ;Done yet?
        JNZ   DEC16LOOP2  ;No, loop back

        MSG    ASCII_BLOCK ;Print string

        POP    AX          ;Restore registers
        POP    CX
        POP    DX
        POP    DI
        RET     ;Return to caller

PRINT_BLOCK
ENDP

DISK_PARMS
PROC
        PUSH    AX          ;Save registers
        PUSH    CX
        PUSH    DX

        MOV     AH,08H     ;Get parms function
        MOV     DL,80H     ;Fixed disk zero
        INT    13H        ;Call BIOS

        MOV     HEAD_0,DH  ;Save head count
        MOV     CYL_LOW_0,CH ;Save cyl. count (low 8 bits)
        MOV     CYL_SCT_0,CL ;Save sector count & cyl-hi

        MOV     DL,HF_COUNT ;Get fixed disk count
        CMP    DL,2        ;Are there 2 drives?
        JNE    SKIP81     ;No, skip second drive check

        MOV     AH,08H     ;Get parms function
        MOV     DL,81H     ;Fixed disk one
        INT    13H        ;Call BIOS

        MOV     HEAD_1,DH  ;Save head count
        MOV     CYL_LOW_1,CH ;Save cyl. count (low 8 bits)
        MOV     CYL_SCT_1,CL ;Save sector count & cyl-hi

SKIP81:
        POP    DX          ;Restore registers
        POP    CX
        POP    AX
        RET

DISK_PARMS
ENDP

TEST_ESDI
PROC
;To determine if ESDI drives are installed, two calls to
;INT 15H system services must be made. The first is COH,
;which points to the configuration area. This tells us if
;an extended BIOS data area is available. If so, a call of
;C1H is made which tells us where the extended BIOS data
;area is. Offset 70H into the extended BIOS data area tells
;us how many ESDI drives are present.

```

Figure 1. PACKCOPY Assembler Source Code (Part 5 of 10)

```

        PUSH    AX          ;Save registers
        PUSH    BX

        MOV     AX,ES      ;Save ES register
        PUSH    AX        ;On stack
        MOV     AH,0C0H   ;Get system config parms
        INT    15H       ;Call system services
        JC     N_EXTBIOS  ;If carry set, not PS/2

        MOV     AL,ES:[BX]+5 ;Get feature byte 1
        AND    AL,04H     ;Mask extd. BIOS area flag
        CMP    AL,00      ;Is it zero?
        JE     N_EXTBIOS  ;Yes, no extended BIOS

;Extended BIOS data area exists, get location and test for drives

        MOV     AH,0C1H   ;Get extd. BIOS data seg.
        INT    15H       ;Call system services
        MOV     BX,70H    ;Load offset into BIOS area
        MOV     AL,ES:[BX] ;Get ESDI indicator
        MOV     HF_COUNT,AL ;Save no. of drives installed
        CMP    AL,00      ;Any ESDI drives?
        JNE    GOOD_EXIT  ;If yes, no error

N_EXTBIOS:
        STC              ;Set carry for error

GOOD_EXIT:
        POP    AX        ;Get ES register back
        MOV    ES,AX     ;Transfer back

        POP    BX        ;Restore registers
        POP    AX
        RET

TEST_ESDI
ENDP

DISK_SIZE
PROC
        PUSH    AX          ;Save register

        MOV     AL,HEAD_0  ;Check heads
        CMP    AL,HEAD_1  ;For same value
        JNE    B_SIZE     ;Jump if not the same

        MOV     AL,CYL_LOW_0 ;Check lower cylinders
        CMP    AL,CYL_LOW_1 ;For same value
        JNE    B_SIZE     ;Jump if not the same

        MOV     AL,CYL_SCT_0 ;Check sectors and cyl-hi
        CMP    AL,CYL_SCT_1 ;For same value
        JE     G_SIZE     ;Jump if good

B_SIZE:
        STC              ;Set carry for error

G_SIZE:
        POP    AX        ;Restore register
        RET

DISK_SIZE
ENDP

CALC_BLKS
PROC
;This routine calculates the total number of 32KB blocks
;of data to be transferred from the source to target disk.
;The number of blocks is stored in a variable as a 16-bit
;unsigned integer.

        PUSH    AX          ;Save register

        XOR    AX,AX      ;Zero AX
        MOV    AL,CYL_SCT_0 ;Get cylinder-hi bits

        ROL    AX,1       ;Rotate left
        ROL    AX,1       ;Two times
        MOV    AL,CYL_LOW_0 ;Get low cylinder bits

```

Figure 1. PACKCOPY Assembler Source Code (Part 6 of 10)



```

ADD     AX,1           ;Add 1 to AX for cylinder 0
MOV     CYLINDER,AX   ;Save cylinder count

MOV     AL,CYL_SCT_0  ;Get sector count
AND     AL,3FH        ;Mask off upper bits
MOV     SECTOR,AL     ;Save sector count

MOV     AL,HEAD_0     ;Get head count
ADD     AL,1          ;Add 1 for head 0
MOV     HEADS,AL      ;Save head count

MOV     AL,SECTOR     ;Get sector count
CBW                     ;Convert byte to word

MUL     CYLINDER      ;Multiply by cylinder count
DIV     DIVISOR       ;Divide by sectors/32KB blk.
MOV     TEMP0,AL      ;Save quotient
MOV     TEMP1,AH      ;Save remainder

MOV     AL,TEMP0      ;Get temp
CBW                     ;Convert byte to word
MUL     HEADS         ;Multiply by heads
MOV     BLOCKS,AX     ;Save no. of blks to transfer

POP     AX            ;Restore register
RET

CALC_BLKS  ENDP

UPDATE_POINT  PROC

;This routine updates the sector/head/cylinder variables used by the
;read/write block routines. Each time a block is transferred, this routine
;is called to increment the pointers by the block size of 32KB so the next
;block of data can be transferred.

        PUSH     AX           ;Save register

        MOV     AL,DIVISOR    ;Get divisor of 64
        CBW                     ;Convert byte to word
        DIV     SECTOR       ;Divide by max sectors
        MOV     TEMP0,AL      ;Save quotient
        MOV     TEMP1,AH      ;Save remainder

        MOV     AL,TEMP1      ;Get no. of sector increments
        ADD     AL,SECTOR_I   ;Add to current sector pointer
        CMP     AL,SECTOR     ;Compare against max sector
        JL     SKIP_SCT      ;If lower, skip adjustment
        SUB     AL,SECTOR     ;Subtract max sectors
        MOV     BL,HEADS_I    ;Get current head value
        ADD     BL,1          ;Add one
        MOV     HEADS_I,BL    ;Store back

SKIP_SCT:

        MOV     SECTOR_I,AL   ;Else, store new pointer

        MOV     AL,TEMP0      ;Get quotient
        ADD     AL,HEADS_I    ;Add in current headS
        CMP     AL,HEADS     ;Compare against max
        JL     SKIP_HEAD     ;If lower than max, skip
        SUB     AL,HEADS     ;Subtract max heads
        MOV     HEADS_I,AL   ;Store new value
        MOV     AX,CYLINDER_I ;Get current cylinder
        ADD     AX,1          ;Add one
        MOV     CYLINDER_I,AX ;Store new value
        JMP     SKIP_HEAD1    ;Skip around

SKIP_HEAD:

SKIP_HEAD1:

        MOV     HEADS_I,AL   ;Store new value

        POP     AX           ;Restore register
        RET

```

Figure 1. PACKCOPY Assembler Source Code (Part 7 of 10)

```

UPDATE_POINT  ENDP

XFER_POINT  PROC

;This routine transfers the updated pointers to the
;pointers used by the block read/write routines.

        PUSH     AX           ;Save register

        MOV     AL,HEADS_I    ;Get current head value
        MOV     HEAD,AL       ;Store it

        MOV     AL,SECTOR_I   ;Get current sector
        MOV     CYL_SCT,AL    ;Update pointer

        MOV     AX,CYLINDER_I ;Get current cylinder
        MOV     CYL_LOW,AL    ;Store cylinder low

        MOV     AL,0          ;Zero AL register
        ROR     AX,1          ;Rotate AX
        ROR     AX,1          ;Twice
        OR     AL,CYL_SCT     ;OR in sector value
        MOV     CYL_SCT,AL    ;Store new pointer

        POP     AX           ;Restore register
        RET

XFER_POINT  ENDP

MAKE_LC     PROC

        CMP     AL,'A'        ;Below upper case?
        JB     LOWER         ;If yes, jump
        CMP     AL,'Z'        ;Above upper case?
        JA     LOWER         ;If yes, jump
        OR     AL,20H        ;OR in bit 5

LOWER:

        RET

MAKE_LC     ENDP

GET_PARM    PROC

        MOV     CH,00         ;Zero high order
        MOV     CL,PARM       ;Get no. of characters in PSP
        CMP     CL,00         ;No parms?
        JE     PARM_ERROR     ;Yes, error
        MOV     DI,00         ;Set index count to zero
        INC     DI            ;Bypass first space
        PARM_LOOP:
        INC     DI            ;Bypass first character
        MOV     AL,PARM+[DI]  ;Get parm character
        MOV     PARM-2+[DI],AL ;Place into buffer
        LOOP   PARM_LOOP     ;Continue until done
        RET                 ;Return to caller

        PARM_ERROR:
        STC                 ;Set carry for error
        RET                 ;Return to caller

GET_PARM    ENDP

PARSE_LINE  PROC

        MOV     DI,00         ;Set index to zero
        DEC     DI            ;Decrement for loop
        CALL   SKIP_SPACE    ;Skip spaces in parm string
                                ;and get character
        JC     PARSE_ERROR    ;If carry set, error

        ;Valid parm character found (source drive)

SRC_FILE:

        AND     AL,0FH        ;Mask for binary
        ADD     AL,80H        ;Add 80H for fixed disk
        MOV     SOURCE_D,AL   ;Move into source drive spec

        INC     DI            ;Increment index

```

Figure 1. PACKCOPY Assembler Source Code (Part 8 of 10)

```

MOV AL,PARM+[DI] ;Get char. again, reset flags
CMP AL,20H ;ASCII space?
JNE PARSE_ERROR ;If not, exit with error

;Source drive specified, check for target

DEC DI ;Decrement for loop
CALL SKIP_SPACE ;Skip spaces

JC PARSE_ERROR ;If carry set, error

;Valid parm character found (target drive)
TGT_FILE:
AND AL,0FH ;Mask for binary
ADD AL,80H ;Add 80H for fixed disk
MOV TARGET_D,AL ;Place into buffer

INC DI ;Up index to parms
MOV AL,PARM+[DI] ;Get character again and
;reset flags
CMP AL,0DH ;End of string?
JNE PARSE_ERROR ;If not, exit with error
PARAM_END:
CLC ;Clear carry for no error
RET ;Return to caller
PARSE_ERROR:
STC ;Set carry for error
RET ;Return to caller
PARSE_LINE ENDP
SKIP_SPACE PROC

;This routine skips over the spaces that
;delimit the parameters

GET_NEXT:
INC DI ;Pointer to parm line
MOV AL,PARM+[DI] ;Get first character
CMP AL,0DH ;End of parm string?
JE LINE_END ;Yes, return with error
CMP AL,20H ;Delimiter?
JE GET_NEXT ;Yes, skip and get next
RET ;Return to caller

LINE_END:
STC ;Set carry for error
RET ;Return to caller
SKIP_SPACE ENDP
SHOW_MSG PROC
PUSH AX ;Save register

MOV AH,09H ;DOS function code for print
INT 21H ;Call DOS

POP AX ;Restore register

RET
SHOW_MSG ENDP

;Messages used are below:
ABORT DB 13,10,10,'PACKCOPY Aborted.',13,10,'$'
ASCII_BLOCK DB ' ' ;5 spaces
ASCII_BLOCK_1 DB ' ' blocks of 32KB to transfer.',13,10,10,'$'
ASCII_COUNT DB ' ' ;5 spaces
ASCII_COUNT_1 DB ' ' 32KB blocks transferred.',13,'$' ;2 spaces

```

Figure 1. PACKCOPY Assembler Source Code (Part 9 of 10)

```

B_PARAMS DB 'ERROR: Bad or missing parameters, '
DB 'enter PACKCOPY <enter> for instructions.'
DB 13,10,'$'
BAD_COUNT DB 'ERROR: two fixed disks not available.',13,10,'$'
BAD_DRIVES DB 'ERROR: drives are different size.',13,10,'$'
CONFLICT DB 'ERROR: source and target numbers '
DB 'cannot be the same.',13,10,'$'
CONTINUE DB 'Do you want to continue? (Y/N) ', '$'
DATA_XFER_CMP DB 10,10,'Fixed disk data transferred.',13,10,'$'
DRV_OKAY DB 'Drives are the same size with', '$'
INSTRUCTIONS DB 13,10
DB 'PACKCOPY Version 1.00',13,10
DB 'Fixed disk mirror-image utility',13,10
DB 'Written by K.E. Maier 1988',13,10,10
DB 'To use enter: PACKCOPY 0 1 <enter>',13,10
DB ' or: PACKCOPY 1 0 <enter>',13,10,10
DB 'Where: ',13,10
DB ' PACKCOPY 0 1 copies drive 0 to drive 1 '
DB 13,10
DB ' PACKCOPY 1 0 copies drive 1 to drive 0 '
DB 13,10,10
DB 'NOTE: only ESDI disk drives are supported'
DB '13,10,'$'
INTROMSG DB 10,10,10,'PACKCOPY Version 1.00'
DB ' by Kevin E. Maier'
DB 13,10,10,'$'
NOT_ESDI DB 'ERROR: no ESDI disk drives are installed.'
DB '13,10,'$'
OUT_RANGE DB 'ERROR: specified drives out of range. '
DB 'must be 0 or 1.',13,10,'$'
PROCEED DB 13,' ',13,10,'$' ;33 spaces between ' '
R_ERROR1 DB 10,'ERROR: while reading source fixed disk.'
DB '13,10,'$'
RUSURE DB 13,10,'CAUTION!!!!',13,10
DB 'You are about to overwrite all data '
DB 'on target disk!',13,10,'$'
TERMINATE DB 'Program terminated!',13,10
DB 'Possible causes are hardware failure or target '
DB 'disk drive',13,10
DB 'is not properly formatted. Run diagnostics '
DB 'and correct',13,10
DB 'problem before attempting PACKCOPY again.'
DB 13,10,'$'
W_ERROR1 DB 10,'ERROR: while writing target fixed disk.'
DB '13,10,'$'
WARNING DB 'WARNING!!!!!!',13,10
DB 'All data on target disk will be overwritten!'
DB '13,10,10,'$'
WRONG_KEY DB ' Invalid response! Please enter Y or N',13,10,'$'

CODE ENDS
END START

```

Figure 1. PACKCOPY Assembler Source Code (Part 10 of 10)

# New Products

## Hardware

### IBM 4207 Proprinter X24E Model 002

The IBM 4207 Proprinter X24E Model 002 is a functionally enhanced companion to the IBM Proprinter family. Enhancements include increased print speed (240 cps draft mode and 80 cps letter-quality mode at 10 cpi); increased throughput; and Propark, the easy-to-use, one-touch paper parking function. The Proprinter X24E incorporates four-part forms handling, and can print on envelopes. This 24-wire printer is designed for attachment to the IBM PC family, IBM displays, and IBM processors. Supported options include the existing IBM Proprinter Sheet Feed, FontSet, and the RS-232 / RS-422 Serial Interface. The Proprinter X24E is a low to medium usage, narrow-carriage, letter-quality and utility printer.

The 4207 Proprinter X24E Model 002 provides all the functions of the IBM 4207 Proprinter X24 Model 001, plus:

- Increased speed and throughput:
  - 288 cps burst speed in 12-cpi draft mode
  - 240 cps burst speed in 10-cpi draft mode
  - 96 cps burst speed in 12-cpi, letter-quality mode
  - 80 cps burst speed in 10-cpi, letter-quality mode
- Four-part form and envelope capability
- Propark (paper parking)
- 14 KB Print Buffer standard
- Attachment capability for selected IBM system displays
- Enhanced RS-232 / RS-422 Serial Interface

### IBM 4208 Proprinter XL24E Model 002

The IBM 4208 Proprinter XL24E Model 002 is a functionally enhanced companion to the IBM Proprinter family. Enhancements include increased print speed (240 cps draft mode and 80 cps letter-quality mode at 10 cpi); increased throughput; and Propark, the easy-to-use, one-touch paper parking function. The Proprinter XL24E incorporates four-part forms handling, and can print on envelopes. This 24-wire printer is designed for attachment to the IBM PC family, IBM displays, and IBM processors. Supported options include the existing IBM Proprinter Sheet Feed, FontSet, and the RS-232 / RS-422 Serial Interface. The Proprinter XL24E is a low to medium usage, wide-carriage, letter-quality and utility printer.

The 4208 Proprinter XL24E Model 002 provides all the functions of the IBM 4208 Proprinter XL24 Model 001, plus:

- Increased speed and throughput:
  - 288 cps burst speed in 12-cpi draft mode
  - 240 cps burst speed in 10-cpi draft mode
  - 96 cps burst speed in 12-cpi, letter-quality mode
  - 80 cps burst speed in 10-cpi, letter-quality mode
- Four-part form and envelope capability
- Propark (paper parking)
- 14 KB Print Buffer standard
- Attachment capability for selected IBM system displays
- Enhanced RS-232 / RS-422 Serial Interface

### IBM Personal Page Printer II

The IBM Personal Page Printer II is a compact, table-top, 300 dots-per-inch

laser printer that prints at up to six pages per minute utilizing a laser / electrophotography process. It contains an internal controller for printing PostScript (R) documents in the publishing environment, as well as other applications that benefit from the versatility of the PostScript page description language.

The IBM Personal Page Printer II contains a controller card with a 16.67 Mhz Motorola 68000 (R) microprocessor, 2 MB of RAM memory, and 43 resident PostScript typefaces. It is supplied with PC parallel (Centronics), RS-232C serial, and AppleTalk (R) connector ports, providing expanded attachment flexibility.

In addition to printing PostScript data, the laser printer is capable of emulating the IBM Proprinter XL and the Hewlett-Packard LaserJet Plus (R). Also available is an optional 2 MB Memory Expansion feature.

Supplied with each laser printer are a high-speed parallel driver and Microsoft Windows (R) screen fonts. Users who wish to take full advantage of the Windows environment should install the screen fonts. Users of Windows and other applications can improve the printer performance across the Centronics parallel interface by installing the high-speed parallel driver.

PostScript is a registered trademark of Adobe Systems Incorporated; M68000 is a registered trademark of the Motorola Corporation; AppleTalk is a registered trademark of Apple Computer Incorporated; LaserJet Plus is a registered trademark of the Hewlett-Packard Corporation; Microsoft Windows is a registered trademark of Microsoft Corporation.

#### Highlights:

- Up to six pages per minute for letter-size paper
- 300 x 300 dots-per-inch resolution
- Very quiet (52 dBAI while printing, 48 dBAI while idling)

- 150-sheet single input drawer stand-ard
- Two output trays for sequenced or un-sequenced output
- Handles various paper sizes and weights
- Handles envelopes, transparencies and labels
- Initial set of supplies included (toner cartridge and photoconductor unit)
- Internal controller providing Post-Script interpreter, page formatting, and printer font storage
- 2 MB of RAM, field-upgradeable to 4 MB
- Internal diagnostics (at power-up time)
- Flexible connectivity: RS232C, PC parallel, AppleTalk

### **IBM Personal System/2 Model 30 286 (8530-E2R)**

The 8530-E2R is identical to the previously announced 8530-E21, except that the E2R provides a unique 50-key function keyboard instead of the Enhanced Personal Computer Keyboard. This model is intended for use in bank teller applications. The model designation is for ordering purposes only. The 50-key function keyboard is customizable by the user.

### **IBM Trade-In Offering for Qualified IBM and Non-IBM Display Stations**

IBM announces a trade-in credit of \$100 on purchased IBM 3275, 3276, 3277, 3278, 3279, 3178, 3179, 3180, 8775, 5251, 5291, and 5292 display stations when replaced by selected IBM Personal System/2 or RT System models on a one-for-one basis. IBM also announces a trade-in credit of \$50 on non-IBM 3270- or 5250-compatible display stations when replaced by

selected IBM Personal System/2 or RT System models on a one-for-one basis.

This trade-in credit is available to customers who have signed, and have in effect, a Volume Procurement Amendment (VPA) or Special Bid for IBM workstations. The trade-in credit is in addition to all other volume discounts currently in effect for the designated replacement IBM workstation. Special bid customers may also qualify for this program. This trade-in credit is also available to customers through the Customer Fulfillment Option.

This program may be modified or withdrawn by IBM at any time upon written notice.

### **New Part Number for Dual Asynch Adapter/A**

The IBM Personal System/2 Dual Asynch Adapter/A, part number 6451013 feature #1014, replaces the IBM Personal System/2 Dual Asynch Adapter/A, part number 6450347 feature #3033. Greater communications speed can be achieved by using this adapter in conjunction with Operating System/2 Extended Edition Version 1.1. This feature is supported on all models of Personal System/2 8550, 8560, 8570, and 8580 system units.

## **Software**

### **Discover/Education – IBM PC and PS/2 Planning and Installation**

IBM announces the addition of IBM PC and PS/2 Planning and Installation to the Discover/Education family of products. IBM PC and PS/2 Planning and Installation is designed to provide education for support personnel and technical coordinators who are responsible for assisting users in the installation and use of IBM PC and PS/2 hardware through a self-paced, learner-oriented, highly interactive, personal computer-based course. The package focuses on the tasks involved in planning, supporting, installing, and

troubleshooting single and multiple IBM Personal Computers and IBM Personal System/2s. IBM PC and PS/2 Planning and Installation is offered as a licensed program product for customer on-site training, and is also available through IBM Learning Centers for a fee.

#### **Highlights:**

This IBM PC and PS/2 Planning and Installation Discover/Education course offers training and information in the following areas:

- Introduction to IBM personal computing systems
- System unit hardware concepts
- Equipment planning and installation
- Troubleshooting and hardware diagnostics
- IBM Personal Computers and IBM Personal System/2s
- Information for registered IBM technical coordinators

### **Discover/Education – OS/2 Introduction and Implementation**

IBM announces the addition of OS/2 Introduction and Implementation to the Discover/Education family of products. This training package is intended for first-time personal computer users, business professionals, office personnel, and experienced DOS users who require basic, advanced, or support-level knowledge of IBM Operating System/2. OS/2 Introduction and Implementation features a self-paced, learner-oriented concept of interactive personal computer-based training.

OS/2 Introduction and Implementation provides training about the IBM OS/2 base operating systems, including the Presentation Manager, and is designed to teach users of either the Standard Edition or the Extended Edition of OS/2 Version 1.1. When this package is installed with OS/2, the education runs in DOS Compatibility mode. While taking

a Discover/Education course in this environment, a student may switch into a native OS/2 session to practice or use newly learned OS/2 commands, then switch back to the Discover/Education course to continue learning OS/2.

This product is offered as a licensed program for customer on-site training, and is also available through IBM Learning Centers for a fee.

#### Highlights:

- Self-paced interactive personal computer-based training
- Modular architecture, allowing for completion of training modules within 25 to 45 minutes each
- Training paths available for users who require either basic, advanced, or support-level knowledge of OS/2
- Fast path for experienced DOS users within certain modules
- Total of 28 training modules, including:
  - Using OS/2 Commands
  - Using OS/2 Presentation Manager
  - Using OS/2 System Editor
  - Configuring OS/2
  - Working with the OS/2 Print Spooler
  - A description of OS/2 system management concepts

### Discover/Education – OS/2 Database Manager Introduction and Implementation

IBM announces the addition of OS/2 Database Manager Introduction and Implementation to the Discover/Education family of products. This training package is intended for users who require basic, advanced, or support-level knowledge of the IBM Operating System/2 Extended Edition Database Manager. This training package imple-

ments a self-paced, learner-oriented concept of interactive personal computer-based training.

OS/2 Database Manager Introduction and Implementation provides training about the OS/2 Extended Edition Version 1.1 Database Manager. When this package is installed with OS/2, the education runs in DOS Compatibility mode. While taking a Discover/Education course in this environment, a student may switch into a native OS/2 session to use the Database Manager, then switch back to the Discover/Education course to continue learning about OS/2 Database Manager.

This product is offered as a licensed program for customer on-site training, and is also available through IBM Learning Centers for a fee.

#### Highlights:

- Self-paced interactive PC-based OS/2 education for Database Manager
- Modular architecture, with each module designed for 25 to 45 minutes
- Training paths available for users requiring either basic, advanced, or support-level knowledge of OS/2 Database Manager
- Total of 25 training modules, including:
  - Using Query Manager to create simple queries
  - Designing a relational database
  - Using Query Manager to create tables
  - Using Query Manager to retrieve and edit data
  - Using Query Manager to customize a report
  - Using Query Manager to build procedures and customize menus and panels
  - Using Structured Query Language (SQL)

### IBM KEE (R)

IBM Knowledge Engineering Environment (R), IBM KEE (R), a knowledge-based systems development tool, is designed to build large, complex knowledge-based applications. It offers advanced artificial intelligence function through a flexible and expressive frame language, inheritance, object-oriented programming, interactive graphics, and a full range of reasoning and analytical tools. IBM KEE provides a rich graphics-oriented environment for editing and debugging applications, and offers features for integrating completed applications with other information systems. IBM KEE is a version of KEE modified to run in an MVS/XA environment on IBM Common LISP for MVS. It is upward-compatible with all versions of KEE 3.1.

KEE is a registered trademark of IntelliCorp, Inc., Mountain View, California. IBM KEE is IBM's version of IntelliCorp's KEE system. Knowledge Engineering Environment is a trademark of IntelliCorp, Inc. IBM Knowledge Engineering Environment is IBM's version of IntelliCorp's system.

#### Highlights:

- Large, full-function, hybrid expert system shell
- Forward and backward chaining
- Powerful user-defined graphics interface
- Hypothetical reasoning and truth maintenance
- Agenda mechanism for controlling rule-firing order
- Logic language, TellandAsk (tm), to build and modify knowledge base
- Interactive development, debugging, and explanation facilities
- Object-oriented programming with frames and methods

TellandAsk is a trademark of Intellicorp, Inc.

## IBM Token-Ring Network Bridge Program Version 2.0 Available

The IBM Token-Ring Network Bridge Program Version 2.0 provides connectivity between 4 Mbps and/or 16 Mbps IBM Token-Ring Network LAN segments. Using the Bridge Program, combinations of 4 Mbps or 16 Mbps IBM Token-Ring Networks can be connected into a single logical network. The IBM Token-Ring Network Bridge Program Version 2.0 also provides network management support by forwarding ring and bridge error information to the IBM LAN Manager Version 2.0.

Currently licensed users of IBM Token-Ring Network Bridge Program Version 1.1 can obtain the new function of Version 2.0 for a program upgrade charge. The program upgrade to IBM Token-Ring Network Bridge Program Version 2.0 is available from January 27, 1989 until July 31, 1989. These upgrades will be available through IBM Authorized Personal Computer Dealers and Industry Remarketers – Personal Computer certified to market IBM Authorized Advanced Products. The remarketer will require a Proof of License for each upgrade ordered. The original Proof of License is the colored front cover page (the page inside the hard binder) or the separate Proof of License page in the program package as identified by IBM to the remarketer. Customers may also order upgrades through the IBM branch office using the IBM Personal Computer / System Program USM&S Upgrade and Certification Order Form.

## Enhanced Connectivity Facilities Servers-Requesters Release 1.2

The Enhanced Connectivity Facilities (ECF) Servers-Requesters Release 1.2 provides additional support for VM/XA and supports VM shared file systems, MVS/ESA in System/370 mode and IBM Disk Operating System (DOS) Version 4.00.

### Highlights:

- Support provided for VM shared file systems in VM SP6
- Additional support provided for VM/XA
- Support provided for MVS/ESA in 370 compatibility mode
- Support provided for IBM DOS Version 4.00
- Release 1.2 continues to support Release 1.0 and Release 1.1 functions
- Full-screen interactive host support
- Message saving
- Reconnect support
- Enhanced ECF Requesters installation
- Additional data interchange formats supported

## Numerical Control Postprocessor Generator Execution Library for the Workstation

Numerical Control PostProcessor Generator Workstation Execution Library (NCPG-XLWS) is a licensed program for the IBM RT system and the IBM Personal System/2. NCPG-XLWS executes numerically controlled machine tool postprocessors generated by the IBM Numerical Control PostProcessor Generator. NCPG-XLWS provides manufacturing customers with a standalone workstation solution for manufacturing site execution of postprocessors generated at a host location.

### Highlights:

- Operates under the AIX or Operating System/2 environment
- Remote site execution of host-generated NCPG postprocessors
- Reads APT-AC cutter location files (CLFILE) or Automatically Programmed Tool (APT) source

generated by IBM CAD / CAM systems

- Supports execution features of NCPG Release 2.0

## Exploring Measurement, Time, and Money – Level 1 Version 1.00

Exploring Measurement, Time, and Money – Level 1 Version 1.00 provides exploratory, structured, and construction activities for the content areas of measurement, time, and money. This program is designed for students working at a kindergarten through second-grade level.

### Highlights:

- Creates a self-paced, highly interactive learning environment for instruction in kindergarten through second-grade levels
- Relates topics of time, money, and measurement to events, objects, and processes that students encounter in their world
- Uses digitized human speech to provide audible instructions, demonstrations, hints, and feedback to eliminate dependence on reading skills
- Motivates learners through colorful PS/2 graphics that are integral to the educational activities
- Supports the IBM Mouse and uses colorful icons to provide further ease of use for young learners
- Provides activities for student learning and tools for teacher use
- Provides a varied combination of exploratory, structured, and constructive activities
- Permits teachers to control the activities the student will see and to set the difficulty level assigned. In an IBM network system, student and group performance reporting is available.



“The ECHO message can be whatever remark is correct. (page 31)

“The primary driving force behind the design of the IBM Personal System/2 Model 30 was cost. (page 1)

“Conversion of current DOS programs into OS/2 deserves careful planning. (page 10)

“Fixed disk seek time is nonlinear. (page 45)

“Virtual storage has important implications for the capacity planner. (page 11)

“Discover/Education can provide cost-effective training. (page 38)

“The display can show from 16 to 64 shades of gray. (page 5)

“The option cards were mounted horizontally, for many reasons. (page 2)

“The Shell comes with a few things already in its main group. (page 22)

“OS/2 will swap segments to disk in order to free memory space. (page 17)

6325-5002-00

