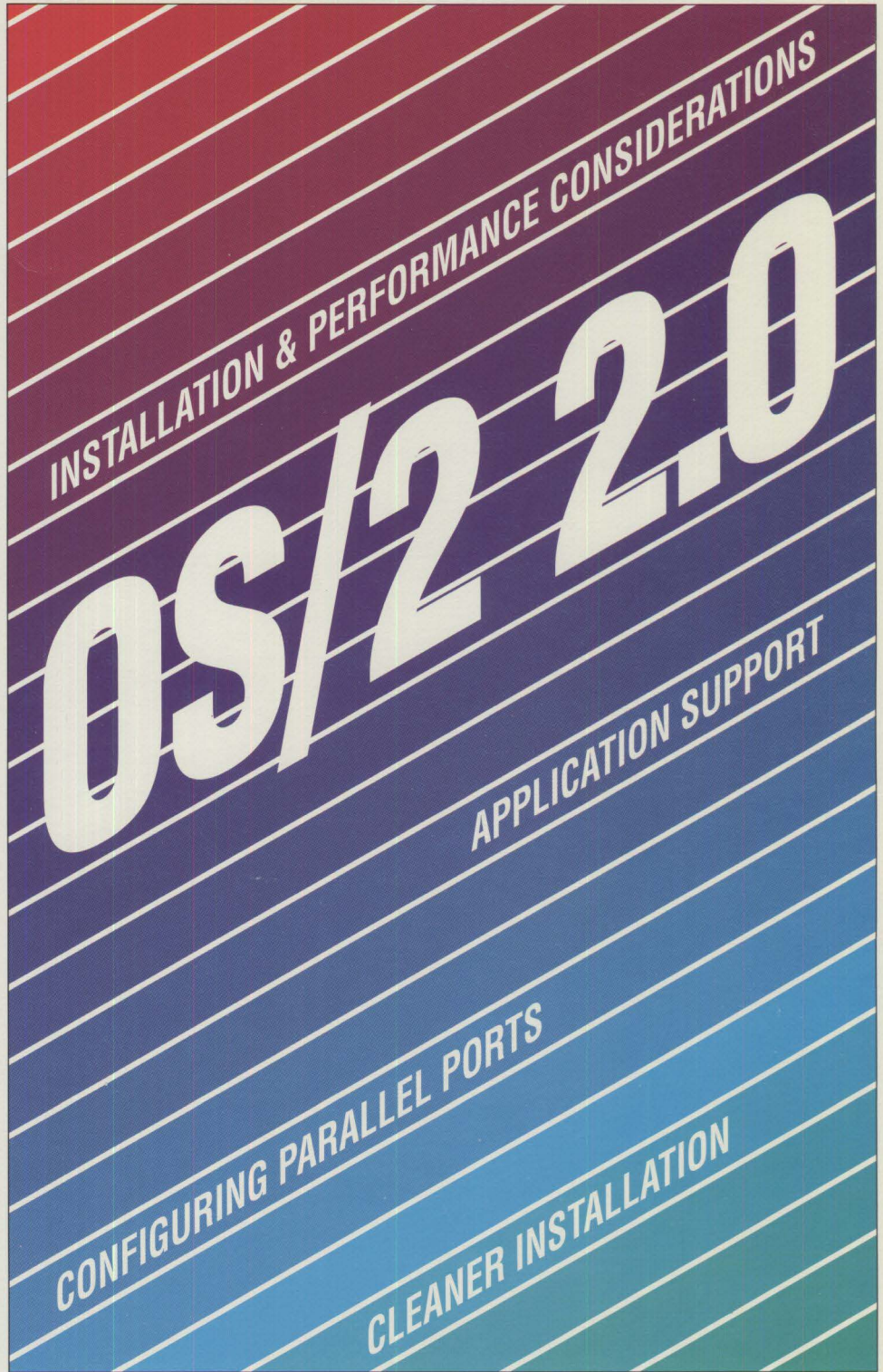


October 1992

# Personal Systems



IBM Personal Systems Technical Solutions



\$12.00

*IBM Personal Systems Technical Solutions* is published quarterly by IBM United States technical support center, International Business Machines Corporation, Roanoke, Texas, U.S.A.

Editor	Libby Boyd
Communications	Elisa Davis
Cover Design	Corporate Graphics
Illustrator	Bill Carr
Publication Services	The TDA Group
Typesetting & Diagrams	Genesis Publications
Manager	Bill Hawkins

To correspond with *IBM Personal Systems Technical Solutions*, please write the editor at:

IBM Corporation  
Internal Zip 40-A2-04  
One East Kirkwood Blvd.  
Roanoke, TX 76299-0015

To subscribe to this publication, call 1-800-551-2832. IBM employees should order through SLSS using order form number GBOF-1229.

Copying or reprinting material from this magazine is strictly prohibited without the written permission of the editor. 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.

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.

Some of the information in this magazine concerns future products, or future releases of products currently commercially available. The description and discussion of IBM's future products, performance, functions, and availability are based upon IBM's current intent and are subject to change.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not imply giving license to these patents.

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.

© Copyright 1992 by International Business Machines Corporation



Printed on recycled paper

# Contents

## Hardware

- 1** Exploring File Server Performance
- 6** PS/2 3.5-Inch Rewritable Optical Drive
- 9** Programming the XGA Video POS Registers
- 14** Video Monitoring on Personal Computers
- 21** Memory Address Space

## Software

- 27** OS/2 2.0 Installation and Performance Considerations
- 41** OS/2 2.0 Application Support
- 56** Cleaner Installation of Applications Under OS/2
- 63** Creating Resizable Pushbuttons
- 66** Configuring Parallel Ports for OS/2
- 71** Performance Characteristics of ES 1.0 Database Manager
- 80** AlertVIEW

## Random Data

- 85** Screen Reader/2
- 88** Little Solutions
- 92** New Products

## **Trademarks**

IBM, PS/2, OS/2, Audio Visual Connection, Micro Channel, Presentation Manager, AT, Personal Computer AT, System/390, Personal System/2, NetView, AIX, and AS/400 are registered trademarks of International Business Machines Corporation.

XGA, Screen Reader, RISC System/6000, PC/XT, Personal Computer XT, SpeechViewer, Independence Series, THINKable, System/370, ES/3090, ES/9000, ESCON, LinkWay, Workplace Shell, PhoneCommunicator, VoiceType, and Ultimedia are trademarks of International Business Machines Corporation.

PostScript is a registered trademark of Adobe Systems Incorporated.  
Apple and LaserWriter are registered trademarks of Apple Computer, Inc.  
ANSI is a registered trademark of the American National Standards Institute.  
Banyan and VINES are registered trademarks of Banyan Systems Incorporated.  
COMPAQ and SystemPro are registered trademarks of COMPAQ Computer Corporation.

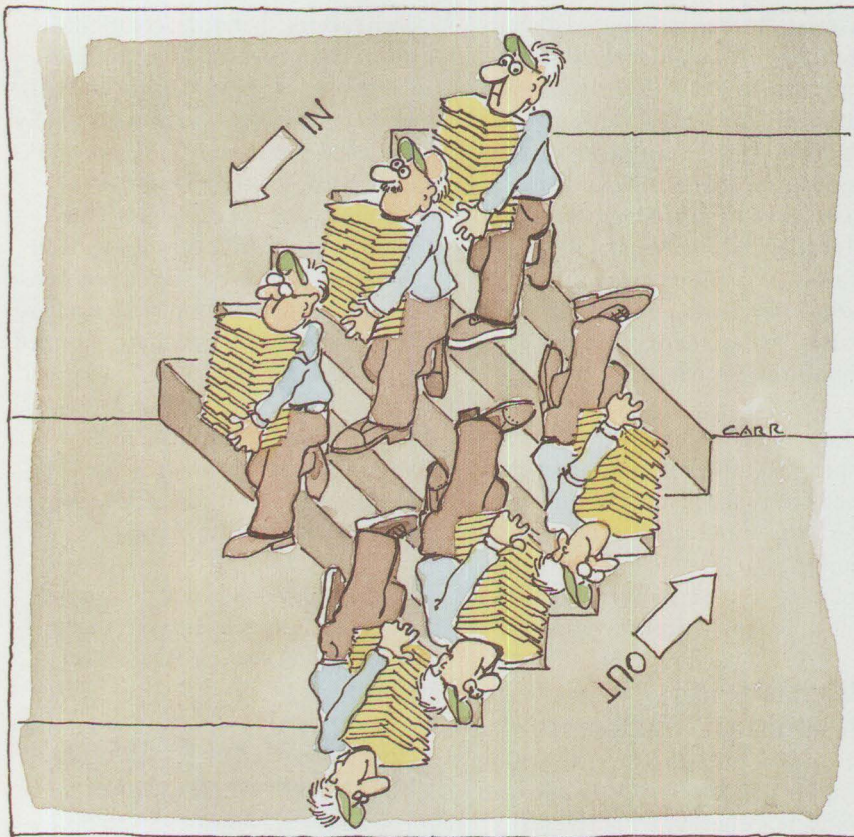
CompuServe is a registered trademark of CompuServe Incorporated.  
ProComm Plus is a registered trademark of Datastorm Technologies.  
Dell is a registered trademark of Dell Computer Corporation.  
DCA is a registered trademark of Digital Communications Associates, Inc.  
Hewlett-Packard and HP are registered trademarks of Hewlett-Packard Company.  
IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers.  
Intel, ActionMedia, and DVI are registered trademarks of Intel Corporation.  
Internet is a registered trademark of Internet, Inc.  
LANSchool is a registered trademark of LAN FAN Technologies.  
Lotus is a registered trademark of Lotus Development Corporation.  
Micro Focus and Micro Focus COBOL/2 are registered trademarks of Micro Focus Limited.

Microsoft and Windows are registered trademarks of Microsoft Corporation.  
Novell and NetWare are registered trademarks of Novell, Inc.  
Express Publisher is a trademark of Power Up Software Corporation.  
AlertVIEW is a trademark of Shany Computers, Ltd.  
Syton Plus is a registered trademark of Sytron Corporation.  
Ethernet is a trademark of Xerox Corporation.  
PC Paintbrush is a registered trademark of ZSoft Corporation.

# Exploring File Server Performance

Gregg McKnight and Avery Lyford  
IBM Corporation  
Boca Raton, Florida

*File servers are specifically designed to handle large volumes of input/output, and sometimes heavy use of the processor. These two requirements conflict — emphasis on one usually degrades the performance of the other. These issues influenced the design of the IBM PS/2® Model 95 486 50 MHz Server. This article discusses the 50 MHz server's dual-bus design, streaming data protocol, processor cache, Error Checking and Correction (ECC) memory, disk subsystem, and network subsystem. It also compares the performance of the IBM PS/2 Model 95 486 50 MHz Server with the performance of other manufacturers' file server systems.*



Local Area Network (LAN) technology enables personal computers, as well as minicomputers and mainframe systems, to share information. In a LAN environment, a powerful file server provides data storage and resource sharing to other computers called *clients*, which are attached to the LAN. A typical PC server is a high-performance 80386- or 80486-based computer with large disk capacity, a high-performance network adapter, and a network operating system.

Most file servers evolved from desktop systems that were originally intended to function as stand-alone computers. These servers have performance problems. File servers must manage an enormous number of Input/Output (I/O) requests, in contrast to most stand-alone systems, which perform relatively little I/O. The evolution in processor speed from the 8088-based IBM Personal Computer to today's 80486-based systems has been accompanied by more than a tenfold increase in processor performance. However, the performance of disks, network adapters, and I/O buses has not kept pace with processor performance. As a result, most desktop systems are not well suited for file-serving applications.

There is a major difference in the requirements of database servers and file servers. A file server must move large amounts of data rapidly, whereas a database server must have a processor that can manipulate large tables of records and indexes. Database servers stress the server's processor, whereas file servers place more stress on the network. Both stress the disk subsystems.

Besides being used as file servers, personal systems are used as database, electronic mail, and communication servers. In these client/server environments, application processing is divided between the client workstation and the server. For example, rather than requesting an entire file in order to update only one record, a database client requests only the required record. The server interprets this request, retrieves the record, and passes it to the client. The server is, therefore, responsible for record searching. This increases the utilization of the server's disks and processor, which in turn decreases LAN traffic.

### Server Performance Requirements

Developing a system that can handle many I/O operations, while providing high-performance processing and enhanced reliability, presents several complex design problems:

**Availability.** Servers are increasingly being used in mission-critical applications, where down time can be very costly.

**Memory.** Servers require vast amounts of memory for the network operating system, File Allocation Tables (FATs), disk cache, and network I/O buffers.

**Reliability.** More memory means a greater chance of a memory failure that could bring down the server. Reliability, therefore, means that the server must continue to function even though it requires significantly more memory than a PC.

File servers are I/O engines. Naturally, a faster processor increases the number of instructions that can be processed in a unit of time. Each I/O request that enters a server has an associated number of instructions that must be executed by its processor. Thus, processor performance can

limit the number of I/O requests that a server can handle. This limits the number of users that the server can support. The effect of a faster processor in a file server is to support more users, *not* to move any single I/O operation through the server faster.

Because file servers are I/O engines, the network and disk subsystems often play a more important role than the processor. Network adapters are the doorway to the server. A slow network adapter impedes server performance because data cannot get into and out of the server fast enough. The disk subsystem empties and fills memory to satisfy network adapter requests for data. A slow disk subsystem reduces the useful capacity of the server system. When the disk cannot write data rapidly, memory fills quickly with cached write data. When this happens, there is no room in memory to handle data being read from the disk into the cache. Read cache hit rates begin to degrade as more requests have to be serviced directly from disk. As additional requests enter the server, performance continues to decline because fewer requests can be serviced from the disk cache, and most requests go directly to the disk subsystem. At this point, all client I/O requests must wait for the slow disk subsystem to read and write information. As a result, the overall performance of the server is significantly reduced.

Ideally, an efficient server should have a processor that offers performance matched to the number of users and the types of I/O requests performed. Furthermore, fast network and disk subsystems are critical for moving data through the server without delay.

### Busmaster I/O Performance Interference

High-performance servers typically employ busmaster network and disk adapters. A busmaster adapter is advantageous because it can move

information to and from memory on its own behalf, without involving the system processor. In contrast, a non-busmaster adapter requires the processor to coordinate the data transfer. The ability of a busmaster to perform unassisted data transfers improves the efficiency of a server and enables it to support an increased workload, additional users, or both.

As busmaster adapters handle larger quantities of information, an interesting phenomenon occurs. As the busmaster uses more memory bandwidth, it reduces the processor's ability to access memory during instruction fetch cycles. For example, in some systems, busmasters that use 50% memory bandwidth cause a 50% decrease in system processing power. This means that the processor is spending 50% of its time waiting to access memory, so that it can sustain only half of its original instruction processing performance.

Two factors, therefore, contribute to increased processor utilization and increased file server workload. First, an increase in I/O operations typically causes an increase in processor workload, because each I/O operation requires some processing. Second, the busmaster I/O interferes with the processor's ability to execute instructions. The more information handled or the more I/O operations performed by the busmaster adapter, the greater the amount of interference that the processor encounters as it tries to access the same memory space. The result is that server capacity decreases in poorly designed systems.

Figure 1 shows how the processor and I/O adapters compete for memory access in conventional server systems.

### IBM PS/2 Model 95 486 50 MHz Server Performance

The IBM PS/2 Model 95 486 50 MHz Server conquers the I/O band-

width interference problem with a unique system design. This design provides a "dual bus" – independent memory paths for the processor and busmaster devices. Therefore, the performance of the new design (shown in Figure 2) is significantly improved over the performance of conventional personal computers used as servers. Under heavy workloads, the PS/2 Model 95 486 50 MHz Server offers superior performance, while the performance of most PC servers would severely degrade.

In addition to its dual-bus capability, the 50 MHz server's performance is also enhanced because it supports the Micro Channel® 40 MB streaming data protocol. Busmaster adapters designed to support the 40 MB streaming data protocol greatly improve data throughput. For example, the IBM 16/4 Token-Ring Busmaster Adapter/A installed in a PS/2 Model 80-311 can transfer data at a rate of up to 6.6 MB/second over the Micro Channel. The same token-ring adapter in the 50 MHz server can transfer data up to 20 MB/second. A 32-bit token-ring adapter with streaming data support can transfer data up to 40 MB/second.

A processor cache is standard in the 50 MHz server. The cache is required in the advanced design of the memory controller in the 50 MHz server and improves processor efficiency up to 2.24 times.

A new 32-bit Direct Memory Access (DMA) controller further improves system efficiency by supporting the Subsystem Control Block (SCB) architecture. By defining the interface between two communicating devices within a PS/2 system, the SCB architecture improves the efficiency of software drivers that must activate the DMA controller. SCBs enable a consistent interface for device programming and communi-

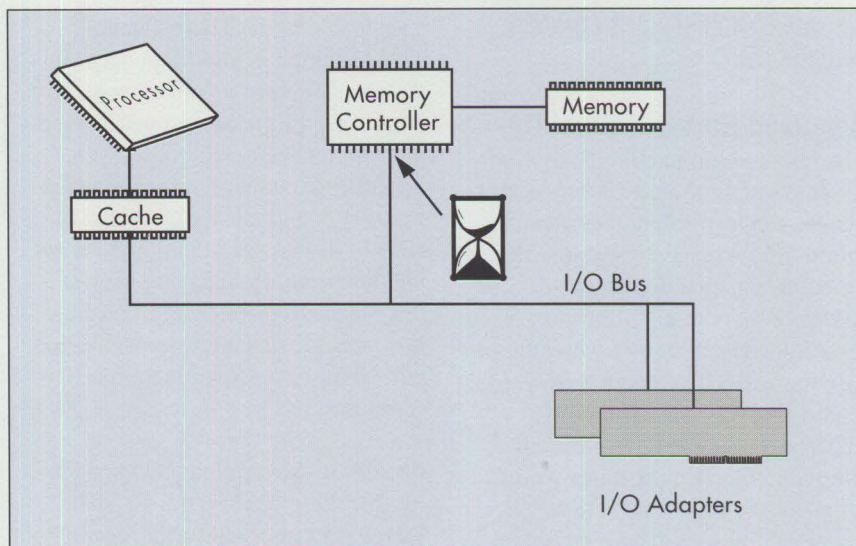


Figure 1. Processor and I/O Adapter Contention for Memory Access

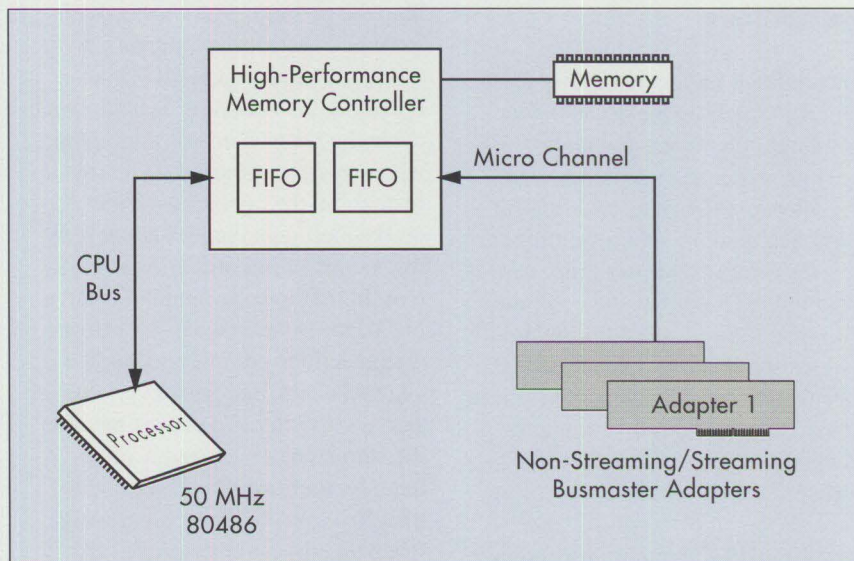


Figure 2. Improved I/O Performance with Dual-Bus Technology

cation between various devices and system memory.

Currently, the most popular use for the system DMA facility in most PC systems is to support floppy diskette I/O operations. Most popular adapters use programmed I/O or busmaster data transfers. Thus, the actual improvements seen by the 32-bit DMA controller will vary depending on the application.

Standard features of the PS/2 Model 95 486 50 MHz Server include 16 MB of Error Checking and Correction (ECC) memory, a 2.88 MB 3.5-inch diskette drive, 256 KB of Level 2 cache, a new Small Computer Systems Interface (SCSI) controller designed to improve client throughput up to 23% over older SCSI adapters, a 32-bit Extended Graphics Array (XGA™) busmaster adapter,

and either a 400 MB or 1 GB SCSI hard drive.

### Increased Server Reliability

Customers are demanding increased server system reliability. Many server outages can be attributed to memory failures. Servers have substantially more memory than most desktop systems. The potential for memory-related outages increases when there is more memory, so servers are particularly susceptible to memory failures. To alleviate this problem, additional reliability features were designed into the 50 MHz server's processor complex. These features include support for both parity memory and ECC memory, although only one kind of memory can be used in a system at a time.

Conventional servers have only parity memory. *Parity memory* can detect only single-bit errors, and all processing stops when an error is detected. *ECC memory* also detects single-bit errors, but then the ECC circuitry corrects the error and allows processing to continue. Of course, the computer stores the address of the failing memory location in a log. The log tells the system administrator that an error has occurred, so the administrator can eventually replace that memory module.

In addition, ECC memory technology can detect rare double-bit errors and some triple- and quadruple-bit errors. The ECC memory checking design of the 50 MHz server has little effect on system performance; throughput is, at most, only a few percentage points lower than with parity checking.

The 50 MHz server also has a facility that checks for data transmission errors over the Micro Channel. This facility, called *Micro Channel Data Parity Checking*, enables busmaster adapters to check for accurate data

transfers over the Micro Channel. This checking is important because the probability of a data error increases as the speed of transfer over the Micro Channel increases. It is interesting to note that the Extended Industry Standard Architecture (EISA) design specification does not include accuracy checking of data that is passed through an EISA bus. It is possible that a bus transfer error will never be detected in an EISA computer.

Besides its Micro Channel Data Parity Checking facility, the 50 MHz server goes one step further by providing a facility called *Synchronous Channel Check*. This facility allows the collection of information that will help the operating system recover from busmaster adapter errors. Synchronous Channel Check lets an operating system know exactly which cycle the busmaster was attempting when an error occurred. With this information, the operating system can make intelligent choices about retrying the operation or halting the system. In previous computers, when a busmaster adapter detected an error, it signaled the computer using the channel check line. However, this signal was activated some time after the actual error occurred. Consequently, there was no choice other than to completely halt processing when an adapter error is detected. No conventional computer system — based on Industry Standard Architecture (ISA) or EISA bus designs — offers recovery from busmaster adapter errors, because it is not part of the architectural definition for those buses.

### Comparison with Other Architectures

Most manufacturers of ISA and EISA systems do not offer system upgrades for ECC memory support, faster I/O bus speeds, and improved

DMA performance, because the integrated circuits required to implement these changes are soldered into the planar board of the systems. Most ISA and EISA system manufacturers claim upgradable processor complexes, but they provide an upgrade path only for the processor and cache. The EISA specification does not even accommodate faster cycles; the current EISA specification limits throughput to 33 MB per second. The IBM PS/2 Model 95 processor complex strategy enables users to upgrade existing machines to support 40 MB per second. Furthermore, the Micro Channel architecture defines rates of 80 MB and 160 MB per second.

All critical integrated circuits of the IBM PS/2 Models 90 and 95 are located on the processor complex. This gives these PS/2 owners a migration path that improves not only processor performance, but also Micro Channel, DMA controller, and memory controller performance and function.

### 50 MHz Server Disk Subsystem Performance

Several advanced features in the design of the disk subsystem in the 50 MHz server have brought a significant improvement in disk subsystem performance. The 50 MHz server's disk subsystem has a much-improved 32-bit caching controller. This device does not support data streaming, but it offers significantly improved network user throughput compared to the 32-bit caching controller found in previous PS/2 systems.

The microprocessor on the new SCSI subsystem executes SCSI microcode faster. Faster adapter memory enables the microprocessor to access memory with fewer wait states. Also, the SCSI interface chip has been replaced with an IBM-developed SCSI chip that reduces SCSI processing overhead. Each of these changes helps the new



32-bit SCSI adapter with cache to process up to 23% more user requests per second. Finally, a SCSI bus terminator has been added on the card, removing the requirement for the bulky external SCSI bus terminators.

### 50 MHz Server Network Subsystem Performance

The IBM 16/4 Token-Ring Busmaster Adapter/A supports the streaming data mode. In streaming data mode on the 50 MHz server, the throughput of a single 16/4 Token-Ring Busmaster Adapter increases up to 17% compared to throughput on the 33 MHz PS/2 Model 95. Performance increases even more when multiple token-ring adapters are used.

Using a 32-bit busmaster streaming data Token-Ring adapter that is available later this year, the 50 MHz server offers up to three times the network throughput as a 33 MHz PS/2 Model 95. This new adapter technology allows the full performance of the 50 MHz server to be realized. Soon, additional adapters such as high-performance Ethernet™, disk, and other network adapters will be available, further enhancing the performance of the 50 MHz server.

### User Benefits

Technical terms, such as improved busmaster throughput, processor performance, ECC memory, and Micro Channel data checking, describe the advanced attributes of the 50 MHz server system. But what do these technical advances mean to users? The best way to illustrate how users benefit from design improvements in the 50 MHz server is to present the results of comparing the 50 MHz server against well-known servers from other manufacturers.

**Comparison with COMPAQ® SystemPro® 486/33:** The SCSI disk subsystem in the IBM Model 95 486 50 MHz Server has a total throughput up to 19.4% greater than the Intelligent Drive Array (IDA) controller in the COMPAQ SystemPro 486/33. Throughput was measured at the client systems while each server was running Novell® NetWare® 3.11 and was under a heavy workload. The IBM and COMPAQ systems had the same number of drives, and no data guarding was used.

**Comparison with Dell® System 450SE:** Compared to the Dell Drive Array, the IBM SCSI disk subsystem in the 50 MHz server yielded up to 326% greater throughput. In this instance, the Dell Drive Array disk subsystem was configured in the composite mode as recommended in the documentation supplied with the Dell System 450SE. When the Dell Drive Array was configured in a fault-tolerant or data-guarding mode, the IBM SCSI disk subsystem provided up to 377% more throughput.

### Conclusion

In LAN file-server environments, the overall system design and balance are of paramount importance. The subsystem components in the PS/2 Model 95 486 50 MHz Server – high-speed I/O bus, error-correcting memory, and dual-bus memory architecture – were designed to respond to these requirements. The 50 MHz server supports large amounts of memory and avoids the down time that results from frequent memory failures. High-performance disk subsystems, with advanced Micro Channel and ECC memory features, increase potential capacity to several hundred simultaneous users. Also, the 50 MHz server

is designed to handle higher bandwidth applications such as multimedia.

*This article is adapted from an article appearing in Issue 2, 1992 of Innovations, an IBM Boca Raton publication for employees.*

*Gregg McKnight is the lead engineer in the LAN Systems Performance department in the IBM Entry Systems Technology laboratory in Boca Raton, Florida. He began working for IBM in 1984 as an associate engineer in the Communication Products Division on the AS/400® communication subsystem. Since 1987, Gregg has been in Entry Systems, working in PS/2 LAN systems performance analysis. He has a BS in physics from the University of Pennsylvania in Edinboro, and an MS degree in electrical engineering from Pennsylvania State University.*

*Avery Lyford is the manager of LAN Systems Performance in the IBM Entry Systems Technology laboratory in Boca Raton, Florida. Avery was hired in 1984 as a junior engineer into the personal computer business unit working on component qualification and reliability. He has continued his work on personal computers and has worked on both workstation and server system performance. Avery has a BS in mechanical engineering from Lehigh University and an MS in computer engineering from Florida Atlantic University.*

# PS/2 3.5-Inch Rewritable Optical Drive

Pedro L. Martinez  
IBM Corporation  
Boca Raton, Florida

*This article discusses the technology of the IBM PS/2 3.5-inch rewritable optical drive, its advantages over other storage device technologies, the applications it supports, and its capacity for storing user data.*

The IBM PS/2 3.5-inch rewritable optical drive provides an excellent platform for supporting a broad range of today's applications. More important, it enables a new generation of possibilities never realized by a personal computer, including the "paperless office" and multimedia.

The PS/2 3.5-inch rewritable optical drive is the first PS/2 optical drive that offers both read and write capability. Because of its high capacity and removability, it is a functional, flexible, secure device.

## Highlights

The highlights of the rewritable optical drive point out the advantages of its leading-edge technology:

- The single-sided removable disk is compact, portable, and securable.
- Its 3.5-inch, half-height form factor is consistent with today's smaller desktop computers.
- The 127 MB read/write media enables the equivalent of a large hard disk to be carried in a pocket or purse.
- The 122 MB read-only media is the ideal capacity and form factor for mass distribution applications.
- Both rewritable media and read-only media can be read by the same drive.

- The Small Computer Systems Interface (SCSI) is the newest industry-standard interface for attaching storage and peripheral devices.
- OS/2® and DOS support give the rewritable optical drive the support of the most popular operating systems today.
- The media, compatible with the American National Standards Institute (ANSI®) industry standard, was designed to industry standards that preserve the interchangeability of media among compliant manufacturers.

## Magneto-Optic Technology

The technology that the PS/2 3.5-inch rewritable optical drive uses to perform read and write operations is called *magneto-optics*. As the name implies, it involves two technologies: electromagnetics and optics. The media is composed of metal particles that are sensitive to magnetic fields and are sandwiched between nonconductive coatings. The entire disk is then enclosed with a plastic casing as on a 3.5-inch diskette.

Figure 1 shows the basic components of magneto-optic technology. They include particles of metal in the media, a laser beam, and an electromagnet.

The state of the components before recording on the media is shown on

the left of Figure 1. New media is shipped with all values at 0. When it is read, it reflects the laser light polarized (rotated) in a counterclockwise direction. This polarity represents the stored value 0.

To perform a write operation, the particles of metal are oriented either perpendicular to the disk surface, which represents binary value 0, or horizontally, representing a binary 1 value. (In Figure 1, to distinguish vertical from horizontal particles, all particles are drawn vertically with directional arrows. Particles that point upward are the true perpendicular particles, with value 0. Particles that point downward represent horizontal particles, with value 1.) To acquire one of these two orientations, a particle is heated by a direct laser beam to 150 degrees Celsius. At that temperature, the particle becomes susceptible to magnetic fields. Next, an electromagnet is applied to the particle. The combination of the laser heat and the electromagnet sets the polarity (the direction) of the particle. The particle then cools and maintains that orientation. This is illustrated in the middle diagram in Figure 1.

The read operation, shown on the right of Figure 1, is much simpler. The laser beam is applied with much lower intensity when aimed at a particle. If the particle reflects the laser light in one direction, it represents the binary value 1; if it reflects laser light in the opposite direction, it represents the binary value 0. The particle shown between the electromagnet and the laser beam in Figure 1 reflects the laser light polarized in a clockwise direction, which represents the value 1.

The write and read operations occur remarkably fast, since the drive is spinning at over 1,800 rotations per minute.

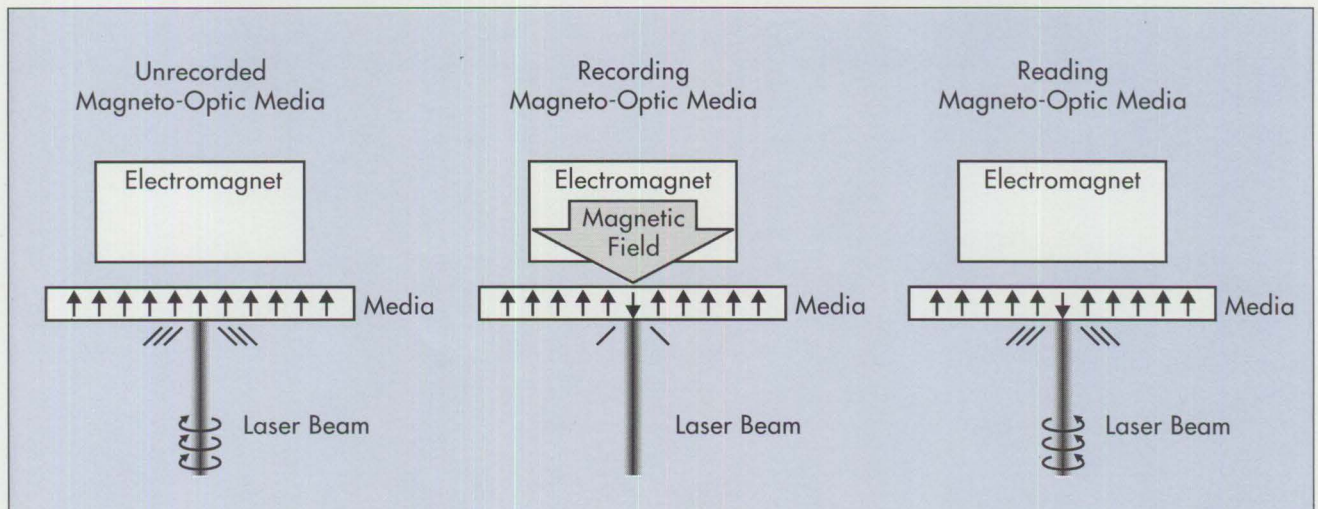


Figure 1. Components of Magneto-Optic Technology

There are several advantages to magneto-optic recording. The data cannot be altered by stray magnetic fields or inadvertent exposure to magnets. Disk crashes are eliminated, because the recording mechanism does not fly as close to the media as with hard disks. Compared to CD-ROM and Write Once/Read Many (WORM) drives, magneto-optics have been tested to one million rewrites. The media is so stable that IBM warrants it for seven years.

### Optical Read-Only Memory

The 3.5-inch rewritable optical drive can read a second type of media, called Optical Read-Only Memory (O-ROM). As with CD-ROM, the data is "stamped" onto O-ROM instead of being written by a drive. Stamping is typically done by third-party vendors who have sophisticated equipment to mass-produce the media. Although this process is costly, the cost is offset because O-ROM media is less expensive than rewritable media, and it is being used in large quantities. Users can choose to replicate their application en masse for large distribution. For fewer copies, they can write the data using the drive itself. In either case, the drive is compatible

with both types of media (O-ROM and rewritable) and both types of data recording (stamped data and written data).

Compared to CD-ROM, O-ROM is the preferred solution for applications that require speed and interactivity. O-ROM is preferred if the drive is used not only for data or program distribution, but also to supplement or back up a hard disk. These applications illustrate the practical, multifunctional capabilities of the rewritable optical drive.

### Applications

The PS/2 3.5-inch rewritable optical drive accommodates several state-of-the-art software applications, such as the following:

- Microfiche replacement
- Paperless office
- Catalog distribution
- Large, flexible server database
- Portable database
- Multimedia presentations
- Hard disk backup
- Additional, large storage

Data or program distribution, data portability, large storage, and saving and restoring are common user requirements. Emerging applications such as multimedia and image storage are also natural for this technology because the technology is faster, denser, removable, faster to replicate, and more secure.

### Comparison with Other Technologies

Figure 2 compares the PS/2 rewritable optical drive with existing technologies used in a variety of applications. The comparisons show that the rewritable optical drive is an excellent multifunctional device that performs tasks typically requiring multiple devices.

### Storage Capacity

How much data can be stored in 127 MB? Here are examples of the amount of data that can be stored on one PS/2 3.5-inch rewritable optical drive:

- 88 high-density diskettes
- 40,000 pages of text
- 1,200 Video Graphics Array (VGA) images

Alternative Media	Optical Advantages	Optical Highlights	Characteristic or Function
Diskette	Denser	88 X More Capacity	Media Distribution Portability
	Faster	1.4 X Average Seek 11 X Data Rate	Save/Restore Multimedia
	Fast Replication	Media Stamping	
CD-ROM	Writable	Read/Write	Media Distribution Multimedia
	Smaller	3.5-inch versus 5.25-inch Drive 3.5-inch versus 4.75-inch Media	
	Faster	6 X Average Access 2.5 X Data Rate	
1/4-inch Tape	Faster	Random Access 100 X Average Seek 11 X Data Rate on Read 3 X Data Rate on Write	Portability Save/Restore Media Distribution
	Fast Replication	Media Stamping	
Multiple Hard Disks	Removable	Only One Drive Required	Mass Storage
	Limitless		Multimedia
	Enhanced Security	Removable Media	

Figure 2. Comparison of the PS/2 3.5-Inch Rewritable Optical Drive with Alternative Media

- 8,000 graphic images (320 x 200 with 256 colors)
- 45 minutes of an IBM Audio Visual Connection® (AVC) multimedia presentation
- 14 minutes of an IBM Digital Video Interactive (DVI®) presentation
- 1.5 hours of high-quality music

In backup and restore applications, the drive can back up a 60 MB hard disk in less than 15 minutes.

### Technology Direction

Rewritable optical technology is poised for growth in capacity and performance. Besides techniques for increasing density and reducing access time, key enhancements will emerge as system support for the technology continues to develop. Advancements in system architectures, increased use of rewritable optical technology as computer subsystems, and state-of-the-art operating systems and applications will accelerate the use of rewritable optical technology.

*Pedro L. Martinez is the product line manager in the marketing organization of the IBM Personal Systems Line of Business in Boca Raton, Florida. Since joining IBM in 1975, Pete's work assignments have included minicomputer system design, advanced technology, computer architecture, and management in robotics, mid-range systems, automotive systems, fault-tolerant computers, personal computers, and technical workstations. He has a BS in electrical engineering from the University of Miami.*

# Programming the XGA Video POS Registers

**Jim Paolantonio**  
**IBM Corporation**  
**Boca Raton, Florida**

*Programmable Option Select (POS) registers are an innovative feature in PS/2 Micro Channel systems that eliminate the need for manual switches on the system board and on adapter cards. As the name implies, these registers must be programmed. This article discusses programming the POS registers on the XGA Display Adapter. It gives details about the contents of each register, and shows how the system puts these contents to use.*

The XGA Display Adapter provides high-resolution graphics (1024 x 768 with 256 colors) along with Video Graphics Array (VGA) mode operation. In the Extended Graphics Array (XGA) graphics mode, the XGA adapter's graphics coprocessor supports hardware drawing assist functions: BitBLT

(Video Bitmap Block Transfer), line drawing, and area fill. These functions are optimized for use with the Microsoft® Windows® and OS/2 Presentation Manager® graphical operating environments.

The XGA video subsystem programming interface has a complex set of

hardware registers with selectable Input/Output (I/O) and memory-mapped addresses. These registers and addresses are as follows:

- 5 Programmable Option Select (POS) registers
- 16 direct I/O registers
- 128 indirect (indexed) I/O registers
- 128 memory-mapped graphics coprocessor registers
- 1 MB video buffer memory-mapped addresses

It is impossible to discuss in a single article how all these registers are programmed. This article concentrates on the functions within the five POS registers used by the XGA Display Adapters.

## POS Register Concept

During Power-On Self Test (POST), each adapter's POS registers are programmed with information about the unique features (the configuration) of



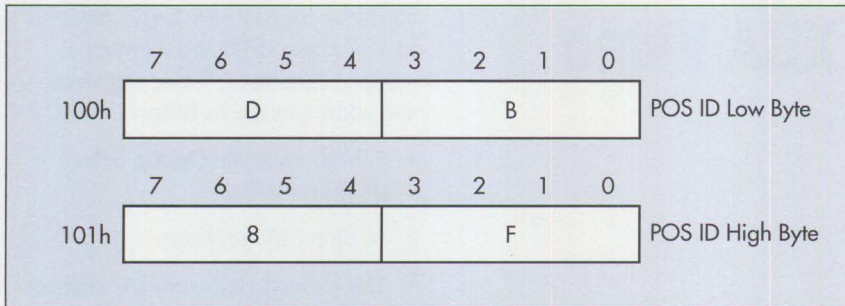


Figure 1. Contents of XGA POS Registers 0 and 1: Subsystem Identification

that adapter. POST reads an adapter's unique POS ID, which is hard-coded into the adapter's POS registers. The POS ID code (similar to a product's bar code) identifies the type of adapter resident in a Micro Channel slot. During setup, the POST process, in conjunction with the system configuration utilities, configures each adapter by programming its POS registers.

The adapters and the system board each have a set of up to eight POS I/O addresses between 100h (where h denotes hexadecimal) and 107h. Depending on the number of unique functions incorporated, a specific adapter may not need to implement all eight POS registers.

The contents of the POS registers can be accessed by using the System Services POS BIOS call INT 15 (software Interrupt 15) with AH = C4h. Although application programs never write to the POS registers, they may need to read an adapter's POS registers. The POS registers can contain pertinent information required to run an application.

### The XGA POS Registers

The XGA video subsystem hardware interface contains its own distinct POS Registers 0, 1, 2, 4, and 5, which respectively have I/O addresses 100h, 101h, 102h, 104h, and 105h. These registers, which are set up only once (during POST), contain information about the locations of the XGA

subsystem registers and the video display buffer.

### POS Registers 0 and 1

XGA POS Registers 0 and 1 contain the low and high bytes, respectively, of the XGA adapter's POS ID. The currently allocated XGA adapter POS ID is 8FDBh. (Additional POS IDs, 8FD8 through 8FDA, are reserved for future versions of XGA.) Figure 1 shows how POS ID 8FDBh is situated in POS Registers 0 and 1.

The system configuration utilities use the XGA adapter's POS ID to locate the corresponding XGA Adapter Description File (ADF). The ADF contains specific information for programming the XGA hardware registers. Figure 2 shows the ADF for the XGA Display Adapter.

### I/O Addressing and Memory-Mapped Addressing

Before describing POS Register 2, it is necessary to explain I/O and memory-mapped addressing.

The Intel® microprocessor architecture has two distinct modes for addressing devices. The first mode is I/O addressing. The real-mode DOS environment has 64 KB of I/O addresses that are available for addressing devices in the system. In this mode, when the processor addresses a device, it uses the Intel assembly language IN instruction to read the XGA I/O registers, and the OUT instruction to write to those registers.

In memory-mapped addressing mode, the processor uses Memory Read and Memory Write instructions to communicate with system memory, diskette drives, and hard disk drives. Memory-mapped addresses total 1 MB in the real-mode DOS environment. The XGA Display Adapter has memory-mapped addresses associated with its video RAM and graphics coprocessor.

### POS Register 2

The primary purpose of XGA POS Register 2 is to provide a card-enable function. This function permits the XGA adapter to be disabled – essentially, turned off – in the Micro Channel slot, so that it does not respond to its assigned I/O addresses and memory-mapped addresses. In addition, POS Register 2 contains the XGA adapter's instance. Because there can be up to eight XGA adapters present in the system, the instance (a number between 0 and 7) distinguishes one XGA adapter from another. POS Register 2 also houses the memory-mapped address range selector for the ROM on the XGA adapter that contains program code. Figure 3 shows the contents of POS Register 2.

#### ENBL

In POS Register 2, when the ENBL bit (bit 0) is a 1, the XGA video subsystem is enabled and responds to all non-POS addresses (that is, to all XGA direct and indirect I/O addresses, and to coprocessor and video RAM memory-mapped addresses). There are 16 direct I/O registers. These registers contain indexes to 128 indirect registers. Direct and indirect registers are discussed at the beginning of this article. When the ENBL bit is a 0, the XGA subsystem responds only to POS register addresses.

#### INST

The INST field, in bits 1 through 3 of POS Register 2, defines an XGA adapter's instance. Each XGA instance has a unique set of direct and indirect

I/O addresses, as well as coprocessor and video RAM memory-mapped addresses. The instance also is used to locate which of the eight blocks of 128 addresses (refer to Figure 4) is assigned to the XGA coprocessor registers.

### ROM Address

The XGA Display Adapter has a Read-Only Memory (ROM) containing the POST code that initializes the XGA registers. This ROM occupies the first 7 KB of an 8 KB memory address block. In POS Register 2, the ROM Address field (bits 4 through 7) specifies which of 16 possible 8 KB blocks of memory-mapped addresses has been assigned to this XGA adapter. The remaining upper 1 KB of address space in this same block is reserved for eight instances of the 128 XGA coprocessor memory-mapped registers.

### Example

To use all these concepts, refer to Figure 4, which shows the correspondence between the XGA instance and the ROM address, and between the XGA I/O register and the video ROM and coprocessor memory-mapped addresses.

The example in Figure 4 assumes that XGA instance 5 (INST=101h) and ROM Address 3 (0011h) have been selected.

In Figure 4, XGA instance 101h yields base address 2150h for the 16 XGA direct I/O register addresses. The 16 registers extend from I/O address 2150h through 215Fh.

At the same time, ROM address 0011h provides a ROM memory-mapped address range of C6000h to C7BFFh. This is the lower 7 KB of the 8 KB address block.

Combining XGA instance 101h and ROM Address 0011h yields a base address of C7E80h for the 128 memory-mapped coprocessor regis-

```

AdapterId 08FDBh
AdapterName "XGA Video Adapter"
NumBytes 4
FixedResources
    pos1=0XXXXXX0b
    pos3=1100XXXXb exec
address 32
Begin Device 03h 02h 00h NoDMA
NamedItem Prompt "Video I/O Address"
    choice "Instance 6: 2160h - 216Fh"
        pos0=XXXX110Xb io 02160h-0216fh
    choice "Instance 7: 2170h - 217Fh"
        pos0=XXXX111Xb io 02170h-0217fh
    choice "Instance 0: 2100h - 210Fh"
        pos0=XXXX000Xb io 02100h-0210fh
    choice "Instance 1: 2110h - 211Fh"
        pos0=XXXX001Xb io 02110h-0211fh
    choice "Instance 2: 2120h - 212Fh"
        pos0=XXXX010Xb io 02120h-0212fh
    choice "Instance 3: 2130h - 213Fh"
        pos0=XXXX011Xb io 02130h-0213fh
    choice "Instance 4: 2140h - 214Fh"
        pos0=XXXX100Xb io 02140h-0214fh
    choice "Instance 5: 2150h - 215Fh"
        pos0=XXXX101Xb io 02150h-0215fh

Help
"The Video I/O address selects a particular I/O address range for
the Display Controller Registers. This field also affects the
exact location of the video coprocessor registers."

NamedItem Prompt "Video Arbitration Level"
    choice "Arbitration level 13" pos1=X1101XXXb arb 0dh
    choice "Arbitration level 12" pos1=X1100XXXb arb 0ch
    choice "Arbitration level 11" pos1=X1011XXXb arb 0bh
    choice "Arbitration level 10" pos1=X1010XXXb arb 0ah
    choice "Arbitration level 9" pos1=X1001XXXb arb 9
    choice "Arbitration level 8" pos1=X1000XXXb arb 8
    choice "Arbitration level 14" pos1=X1110XXXb arb 0eh

Help
"The video subsystem can be assigned any one of the
available arbitration levels 8 through 14. Use the
F5=Previous and the F6=Next keys to change arbitration
level assignments if you are in the 'Change Configuration'
window. Conflicting assignments are marked with an asterisk
and should be changed."

NamedItem Prompt "Video Fairness"
    choice "Fairness On" pos1=XXXXX1XXb
    choice "Fairness Off" pos1=XXXXX0XXb

Help
"Video Fairness indicates whether or not the video
subsystem coprocessor will follow the fairness algorithm
for bus usage. This setting can be changed if you are in
the 'Change Configuration' window by using the
F5=Previous and F6=Next keys."

End

```

Figure 2. Adapter Description File for the XGA Display Adapter

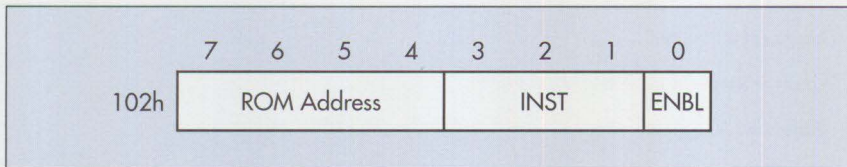


Figure 3. Contents of XGA POS Register 2: ENBL Bit, Instance, and ROM Address

ters. These registers reside within the upper 1 KB of the same 8 KB address block, and they extend from address C7E80h through C7EFFh.

#### POS Register 4

POS Register 4, shown in Figure 5, contains the base address of XGA video memory.

#### Video Memory Base Address

The XGA Display Adapter contains up to 1 MB of video memory. The seven most significant bits of the base address of the XGA memory are defined in bits 1 through 7 of POS Register 4. The three least significant bits of the base address of XGA memory are defined in the three XGA

instance bits in POS Register 2. These two fields of bits are concatenated to define a 10-bit video memory address located on a 4 MB address boundary.

Figure 6 shows how to determine the video memory address. In Figure 6, bits 25 through 31 contain a copy of the information in bits 1 through 7 of POS Register 4. These seven most significant bits of the video memory address are set to binary 4 (0000100b). Also in Figure 6, bits 22 through 24 contain a copy of the information in the XGA instance (bits 1 through 3 of POS Register 2). These three least significant bits of the video memory address are set to binary 3 (011b).

ROM Address		Coprorocessor Register Base Address (in Hex)							
Field	Range (in Hex)	Instance (INST)							
		0	1	2	3	4	5	6	7
0000	C0000 - C1BFF	C1C00	C1C80	C1D00	C1D80	C1E00	C1E80	C1F00	C1F80
0001	C2000 - C3BFF	C3C00	C3C80	C3D00	C3D80	C3E00	C3E80	C3F00	C3F80
0010	C4000 - C5BFF	C5C00	C5C80	C5D00	C5D80	C5E00	C5E80	C5F00	C5F80
0011	C6000 - C7BFF	C7C00	C7C80	C7D00	C7D80	C7E00	C7E80	C7F00	C7F80
0100	C8000 - C9BFF	C9C00	C9C80	C9D00	C9D80	C9E00	C9E80	C9F00	C9F80
0101	CA000 - CBBFF	CBC00	CBC80	CBD00	CBD80	CBE00	CBE80	CBF00	CBF80
0110	CC000 - CDBFF	CDC00	CDC80	CDD00	CDD80	CDE00	CDE80	CDF00	CDF80
0111	CE000 - CFBFF	CFC00	CFC80	CFD00	CFD80	CFE00	CFE80	CFF00	CFF80
1000	D0000 - D1BFF	D1C00	D1C80	D1D00	D1D80	D1E00	D1E80	D1F00	D1F80
1001	D2000 - D3BFF	D3C00	D3C80	D3D00	D3D80	D3E00	D3E80	D3F00	D3F80
1010	D4000 - D5BFF	D5C00	D5C80	D5D00	D5D80	D5E00	D5E80	D5F00	D5F80
1011	D6000 - D7BFF	D7C00	D7C80	D7D00	D7D80	D7E00	D7E80	D7F00	D7F80
1100	D8000 - D9BFF	D9C00	D9C80	D9D00	D9D80	D9E00	D9E80	D9F00	D9F80
1101	DA000 - DBBFF	DBC00	DBC80	DBD00	DBD80	DBE00	DBE80	DBF00	DBF80
1110	DC000 - DDBFF	DDC00	DDC80	DDD00	DDD80	DDE00	DDE80	DDF00	DDF80
1111	DE000 - DFBFF	DFC00	DFC80	DFD00	DFD80	DFE00	DFE80	DFF00	DFF80
XGA "INST" Field		000	001	010	011	100	101	110	111
I/O Base Address		2100	2110	2120	2130	2140	2150	2160	2170

ROM Address 0011

ROM Address Range C6000 - C7BFF

Coprorocessor Base Address C7E80 (Addresses C7E80 - C7EFF)

Instance 5

I/O Base Address 2150  
Addresses 2150 - 215F

Figure 4. XGA I/O Address and Memory-Mapped Address Blocks



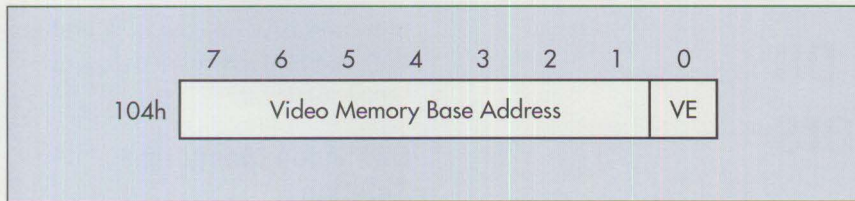


Figure 5. Contents of POS Register 4: Video Memory Base Address

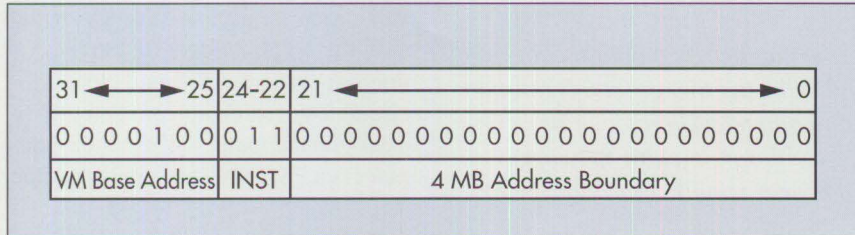


Figure 6. Finding the Base Address of XGA Video Memory

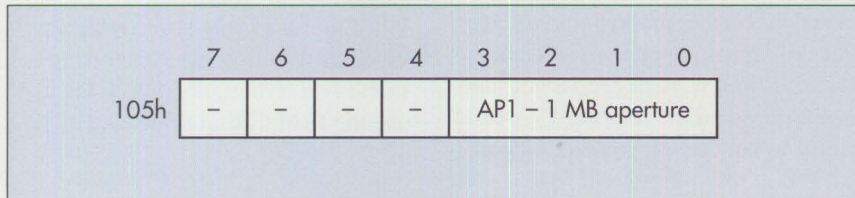


Figure 7. Contents of XGA POS Register 5: 1 MB Aperture

Together, bits 22 through 31 in Figure 6 constitute the 10-bit video memory address. Next, bits 0 through 21 in Figure 6 contain zeros. These represent the 4 MB address boundary. Finally, bits 0 through 31, taken together as one big number, yield the video memory address of 08C00000h. This is a 32-bit address that can be handled only by 32-bit 80386 and 80486 processors (see the article "Memory Address Space" in this issue).

**Video Enable (VE)**

There is a 4 MB aperture through which XGA video memory can be accessed. The *aperture* is a range of system addresses through which an application program can write to, or read from, the video memory. When the Video Enable (VE) bit in POS Register 4 is set to 1, the 4 MB address aperture is enabled; when the

VE bit is 0, the 4 MB aperture is disabled.

**POS Register 5**

The XGA Display Adapter also has a 1 MB aperture (not the same as the 4 MB aperture just mentioned) through which the video memory also can be accessed. As shown in Figure 7, bits 0 through 3 of XGA POS Register 5 define where the 1 MB aperture is located in the system address space.

Figure 8 shows the correspondence between the contents of the four-bit field labeled API in Figure 7 and the address of the 1 MB aperture. In Figure 8, when the four bits in API are set to 5 (0101b), the video buffer can be accessed through the 1 MB aperture located between addresses 00500000h and 005FFFFFh.

Video Memory Aperture  
00500000 - 005FFFFFh

API = 5

API	1 MB Aperture
0000	Disabled
0001	00100000
0010	00200000
0011	00300000
0100	00400000
0101	00500000
0110	00600000
0111	00700000
1000	00800000
1001	00900000
1010	00A00000
1011	00B00000
1100	00C00000
1101	00D00000
1110	00E00000
1111	00F00000

Figure 8. Video Memory 1 MB Aperture Base Address

*Jim Paolantonio is an advisory engineer in visual subsystems within the IBM Entry Systems Technology laboratory in Boca Raton, Florida. His current responsibilities include Very Large-Scale Integration (VLSI) Liquid-Crystal Display (LCD) controller-chip development for portable systems. Jim's previous assignment was XGA video hardware development for the IBM PS/2 Model 90 XP 486. He has also been a member of development engineering teams for the IBM PS/2 Model 80, PC AT®, and PC/XT™. Jim holds BS and MS degrees in electrical engineering from Purdue University.*

# Video Monitoring on Personal Computers

**Geoffrey R. Amthor**  
**Impact Ideas, Inc.**  
**Flowery Branch, Georgia**

*With the IBM video monitoring solution, the gap between digital computers and analog video has been bridged. Now the two great information revolutions of the 20th century – computing and television – can be brought together seamlessly at the desktop with single-cable networkability across entire organizations and at a surprisingly affordable price.*

In most organizations today, desktop video means walking to a conference room and playing a videotape. While you are viewing that tape, you are out of touch with your colleagues, perhaps missing spur-of-the-moment meetings and important phone calls.

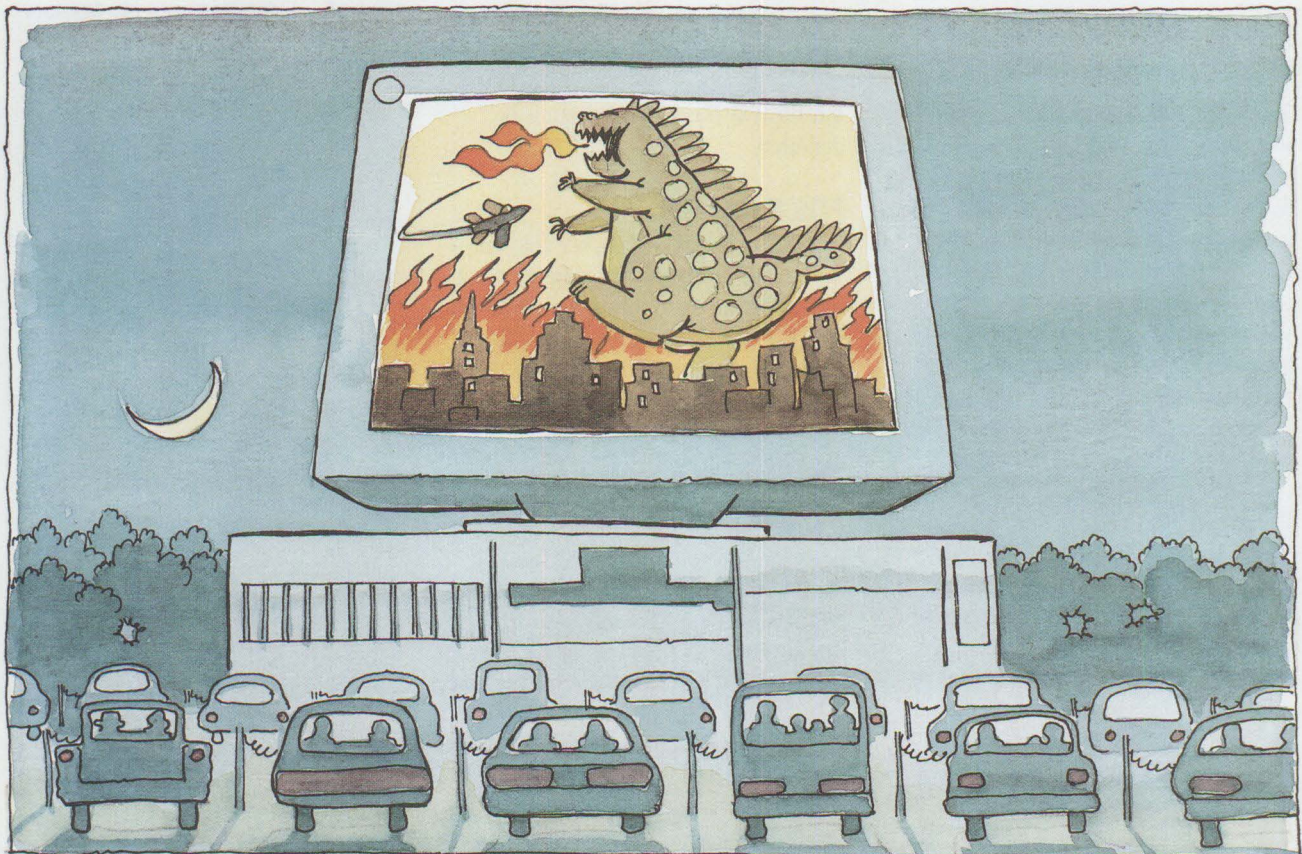
The same situation is true for live video. If you are a financial broker, you want to keep in touch with world events or current market activity. If you are an employee in a large corporation, you want to keep abreast of events aired on the internal TV network. In both cases, you face a choice

of having a TV on your desk or shuttling to a conference room to view broadcasts.

## IBM Video Monitoring Solution

With the IBM video monitoring solution, you can have TV in your desktop computer. Everyone who needs video sources, such as cable TV, broadcast TV, VCRs, videodisc players, internal corporate TV, and closed-circuit TV, can do so conveniently at a desktop computer – whether the computer is a PS/2, PC AT, or compatible.

These video sources can be displayed in full-screen mode or in a window alongside computer applications on standard VGA monitors. IBM's video monitoring solution supports video distribution over token-ring Local Area Networks (LANs) that use the IBM Cabling System (ICS).



Video is one of many applications run at the desktop – with no impact on the performance of digital applications. Because the technology is affordably priced, video monitoring can be deployed across entire organizations. In token-ring installations, more than 70 channels of video can be passed simultaneously over the network.

## PS/2 TV

The key enabling product of IBM's video monitoring solution is IBM PS/2 TV, shown in Figure 1. It is packaged in the slim external enclosure positioned directly beneath the PS/2 monitor in the figure. It contains a 181-channel, cable-ready TV tuner that accepts National Television Standards Committee (NTSC) broadcast signals from a cable TV source or antenna, as well as standard base-band video input from a VCR, laser videodisc player, or video camera. An internal speaker, headphone jack, and a set of video and audio input/output connectors complete the PS/2 TV unit. PS/2 TV is a very affordable enhancement to desktop PS/2, PC AT, and compatible systems<sup>1</sup>.

During installation, the PS/2 display cable is routed from the monitor to the PS/2 TV unit, with a second video cable going from the PS/2 TV unit to the PS/2 system unit. Similarly, the keyboard cable is routed directly to the PS/2 TV unit, with a second keyboard cable routed from the PS/2 TV unit to the standard keyboard connector in the PS/2 system unit. When the PS/2 TV unit is inactive, keyboard and video signals pass through the unit unimpeded.

PS/2 TV is activated through either software or a hot-key combination captured by the unit. When activated, PS/2 TV has standard television controls – channel selection, volume,

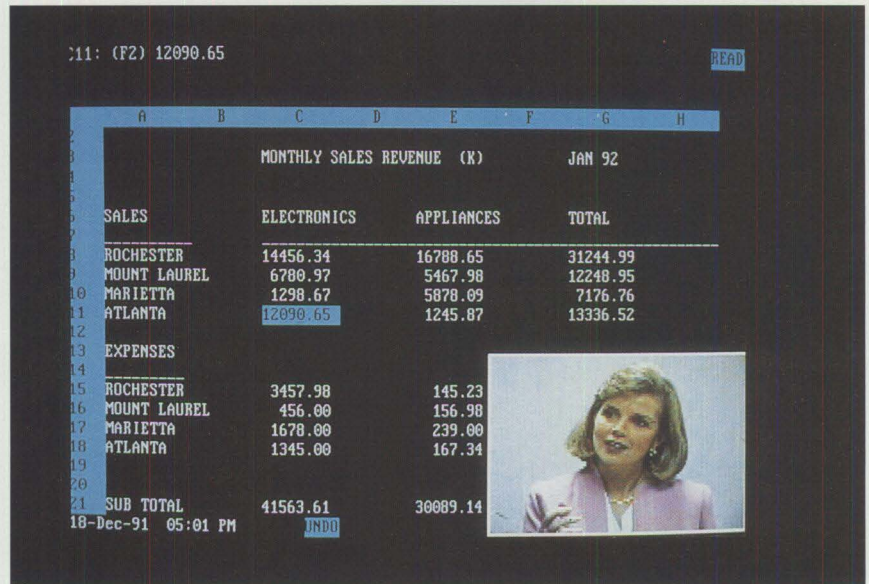


Figure 1. PS/2 TV

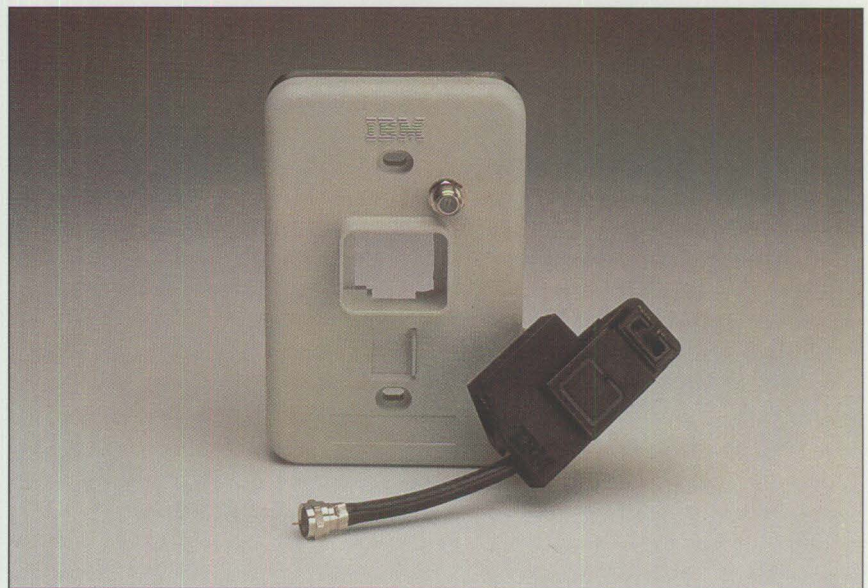


Figure 2. IBM F-Coupler with Faceplate

mute, brightness, contrast, and color. It provides three viewing modes:

- The normal full-screen computer display
- Full-screen television
- A full-screen user application overlaid with a movable Picture-in-

Picture (PIP) TV video image, which uses one-ninth of the screen (In its current version, PS/2 TV supports PIP functions only in VGA display mode; XGA is supported only in full-screen mode.)

Audio is available in any of the modes, and it can be muted. Video received

<sup>1</sup> This article refers to a desktop computer as a PS/2. However, IBM PS/2 TV supports the VGA-enabled IBM PC AT and fully compatible desktop computers. A PC AT and some compatible computers require a keyboard port adapter that is supplied with IBM PS/2 TV.

can be recorded onto a VCR by using PS/2 TV's video- and audio-out capabilities. The keyboard is normally used to control PS/2 TV, but under OS/2 or Windows the functions can be controlled through menu selections as a concurrent application.

Display modes can be changed and television parameters controlled just as with a standard television set. Control is fast and intuitive – a novice can learn in minutes.

Because PS/2 TV comes with its own microcontroller, video places no demands on the PS/2 system unit. Computer applications run as fast as ever.

IBM PS/2 TV offers an affordable enhancement to desktop PS/2s. At the time this article was written, the suggested retail price in the U.S. was \$495<sup>2</sup>.

### Network Distribution System

While PS/2 TV is often used in stand-alone implementations, it also can be used as part of an organization-wide strategy of accessing networked video from desktops.

A basic distribution system can be installed simply by wiring standard coaxial cabling to each PS/2 TV unit in the organization. At the studio headend, several video sources can be distributed on their own programming channel; video source choices include cable TV, satellite, antenna, VCRs, videodisc players, and video cameras. In addition to the video sources, a distribution system is required, typically consisting of a combiner, an RF video distribution amplifier, and a multisplitter.

For organizations with existing or planned token-ring networks using the shielded twisted-pair IBM Cabling

System, a novel approach can take advantage of the ICS cabling and eliminate the need to pull separate coaxial cabling for video signals. This exciting solution is possible because both 4 Mbits and 16 Mbits per second token-ring networks leave approximately 500 MHz of free bandwidth, which is available for video transmissions. In turn, this free bandwidth can easily support over 70 channels of video.

### F-Coupler

To distribute video over a token-ring network, a filtering device called an IBM F-Coupler must be added at each endpoint, as shown in Figure 2. One F-Coupler at the headend merges data signals from the token-ring Multistation Access Unit (MAU) with video, as shown in Figure 3. The F-Couplers at the PS/2 endpoints separate those signals. In small installations, the headend F-Coupler may be located in a studio with the video sources, the video distribution system (combiner, RF video distribution amplifier, and splitter), and the token-ring MAU. In larger installations, the headend F-Coupler would be placed on the token-ring distribution panel in the wiring closet where video and data signals are received.

Because wiring represents a significant expense in a cabling system, the IBM F-Coupler affords an excellent way to reduce the investment required to handle data and video transmissions that previously have used separate cables.

The IBM F-Coupler is a highly affordable distribution solution. At the time this article was written, F-Couplers had a suggested retail price in the U.S. of \$85 each. Each receiving F-Coupler needs a special faceplate (shown in Figure 2). At this

writing, faceplates were available for a suggested retail price of \$6.85.

### Video Monitoring Applications

IBM PS/2 TV enables video feeds from sources such as cable TV, VCRs, videodisc players, and closed-circuit cameras to be displayed on existing PS/2 monitors – eliminating the need for redundant equipment and allowing video to be integrated into normal working environments. Combining both pieces of equipment saves desk space and cost. More important, time is saved because users can quickly switch between video and computer information.

Now you can access a world of video sources. Imagine the applications!

**Broadcast monitoring:** Information sources such as network and cable news broadcasts can be monitored on a workstation. This function is valuable to stockbrokers, financial analysts, journalists, and others who need to keep abreast of world events. A small window displays the live video feed while they continue to perform their normal work routines. To enhance concentration on normal work, video can be delivered silently, or the program can be monitored in audio-only mode. When an item of interest arises in the broadcast, the user can jump into PS/2 TV's dedicated mode, complete with full-screen video and audio. The result is easy access to vital information and improved productivity.

**Business television:** Many corporations today have private video studios that produce internal programming comparable to that of major television networks. Satellite and mobile uplink facilities are readily available and reasonably priced. With the IBM video monitoring solution, even small companies can broadcast inter-

<sup>2</sup> Prices are subject to change. For a limited time, the IBM Multimedia Information Center is offering PS/2 TV direct for \$399. Call (800) 426-9402.

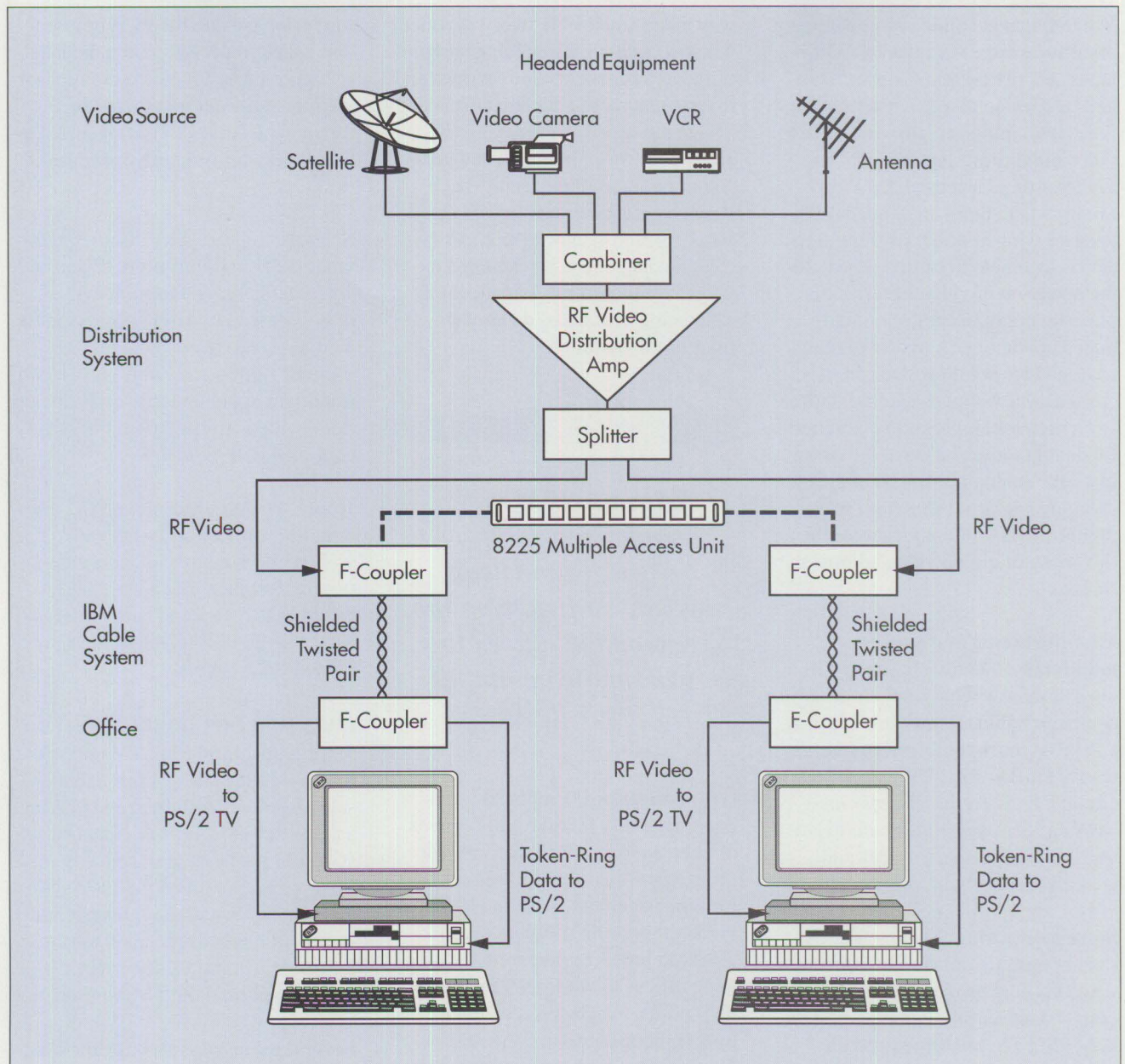


Figure 3. Example of PS/2 TV Distribution System

nal programming to the desktops of their workers – allowing them to work at their desk and still receive vital corporate broadcasts conveniently. Another excellent application is business-to-business television. IBM customers can use the video monitoring solution to view IBM Field Television Network (FTN) telecasts in their own offices.

**Training:** The cost of training employees goes far beyond curriculum materials and trainer fees. For live training, organizations also must consider travel and housing expenses and the loss of employee time from their day-to-day positions. For “canned” training delivered by videotape, students usually leave their desks and go to a conference room or

training facility. With the IBM video monitoring solution, that material can be delivered directly to student desktops. For live training, students can still interact with the instructor by telephone – allowing cost-effective training anywhere. Taped instruction also can be played from a VCR attached directly to the PS/2 TV at the desktop.

**Video library:** While organizations often store large volumes of videotapes and videodiscs, the process of getting a video to a requester can be time-consuming and costly. The IBM video monitoring solution enables the creation of a centralized video library where tapes are loaded for local viewing at desktops. Equipment also is available to further automate the operation of video libraries by providing remote control or automated operations. A hybrid digital/analog video solution also can be created, where video is stored digitally on network servers and converted for analog transmission to the desktop. This analog transmission avoids clogging network bandwidth with digital video, while it preserves the flexibility of digital video editing and storage.

#### **Manufacturing and process**

**monitoring:** Manufacturing supervisors can now work on their desktop computer applications while keeping an eye on the manufacturing line – thanks to IBM PS/2 TV and closed-circuit television. Intelligent video workstations can be programmed to monitor critical safety issues and alert an operator when needed.

**Store monitoring:** Sales clerks can ring up transactions on PS/2 systems while keeping an eye on store operations – thanks again to the pairing of IBM PS/2 TV and closed-circuit cameras.

**Surveillance monitoring:** Multiple PS/2 TV units can be attached to a single PS/2, allowing several surveillance cameras to display within windows on a single monitor. Motion detectors and other system peripherals, such as autodialers, can transform a desktop PS/2 into a comprehensive security system.

**Videoconferencing:** The same infrastructure and equipment used for one-

way video can enable two-way video within a building. By splitting the band of video frequencies into forward and reverse channels, a technician can design the distribution system to allow point-to-point video and voice communication. If the facility has a videoconferencing suite that uses a codec for land-line wide-area videoconferencing, it can be extended to allow an executive to join a conference being conducted on another floor of the building.

*With the IBM video monitoring solution, organizations can broadcast live from the executive offices to workers watching at their desks.*

**Group meetings:** Employee meetings frequently involve large groups of people crowded into a conference room, viewing a videotaped message from management. With the IBM video monitoring solution, organizations can broadcast live from the executive offices to workers watching at their desks. Employees can interact by telephone or video conference link. For those who missed the meeting, a tape can be played for several days from a library. Employees need only select the appropriate channel on their workstations.

**Collaborative workstations:** Multimedia applications and presentations are often created by a team of workers who concentrate on specific parts of the overall project. Presentation files can be created and stored on a token-ring network file server, while video segments can be stored in ana-

log form on a central VCR or laser-disc player/recorder. When the team is supported by the video monitoring solution, all team members can remotely access both analog and digital sources conveniently from their desks.

**Schools:** Schools may discover that video information is more effective when delivered to the students' desks. Instead of displaying video on a large-screen television to a classroom of students, schools can deliver important documentaries and educational videos to classroom PS/2s equipped with PS/2 TV.

**Home:** With a stand-alone PS/2 TV unit, home office and weekend workers do not have to choose between computer-based work and catching the weekend ball game. They can do both, right on their home PS/2 systems.

#### **Installing and Using PS/2 TV**

Setting up a local PS/2 TV unit is a matter of connecting a few cables, selecting a video source, and making a couple of adjustments. Because PS/2 TV is an external device, it is not necessary to open the PS/2 system unit for installation. Instead, the installation process is analogous to installing a home VCR between video inputs and the TV monitor.

Several cables pass through the PS/2 TV unit:

- The keyboard plugs into the IBM PS/2 TV unit, with an output cable connected to the PS/2 system.
- The display cable is routed from the monitor to the IBM PS/2 TV unit, with a cable then connected to the IBM PS/2 system unit.
- The video source – VCR, videodisc player, antenna, cable TV, or closed-circuit camera – is hooked up directly to the IBM PS/2 TV unit.

- A power cable runs from a wall transformer to the IBM PS/2 TV unit.
- There are video- and audio-out connectors in the IBM PS/2 TV unit for the attachment of a VCR or other recording device.

### How PS/2 TV Works

Once cabled and configured, the IBM PS/2 TV unit is always ready. As with any television set, the picture and sound are standing by – waiting for the user to turn on the set, select a channel, and adjust the volume. With PS/2 TV, you make a few other decisions too, such as selecting a viewing size, placement of the video window, and any personal fine tuning. In all cases, PS/2 TV's display overlays the VGA signals from the PS/2 system unit, much like a VCR's on-screen programming instructions overlay the TV's own video display.

The PS/2 TV unit picks up the video signal from a designated source: antenna, cable, VCR, or videodisc. On user command, it then delivers this video to the PS/2 display. There are three display options:

- Full-screen computer application display (PC mode)
- Picture-in-Picture video window, overlaying the active computer application
- Full-screen video display

You can quickly switch among these modes by pressing a key or clicking a mouse.

### How to Control PS/2 TV

The method of turning on the "set," choosing a channel, or making adjustments depends on the operating system. With DOS, a series of keyboard entries is used for volume, channel selection, picture adjustments, viewing mode, and picture location. As each decision is made, a correspond-

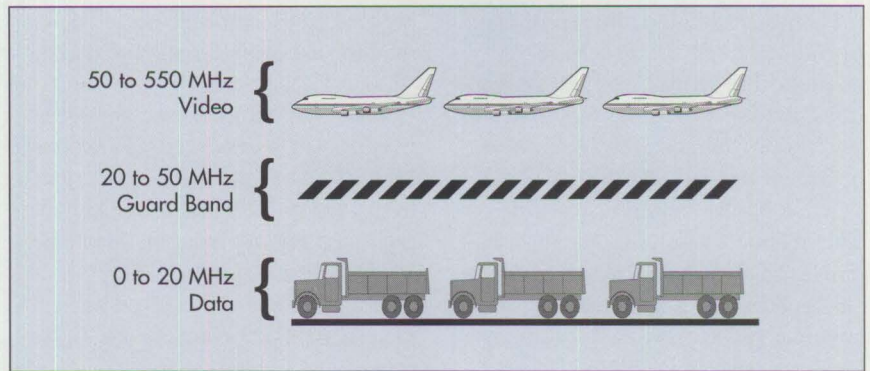


Figure 4. Additional Bandwidth Available with Shielded Twisted Pair

ing on-screen panel highlights and guides interactions. You also can select a timeout for these panels.

When the keyboard is used for these adjustments and actions, it is temporarily dedicated to the PS/2 TV unit. Pressing the Num Lock key twice in quick succession activates this dedicated mode. When you are finished, the Escape key returns control to the PS/2 system unit. No software is required for this keyboard operation.

In a Windows or OS/2 environment, there is an option of using these same key sequences or a series of menus and control panels to perform all the PS/2 TV functions. Windows and OS/2 application software are provided with the PS/2 TV.

### Customizing PS/2 TV

A supplemental Application Programming Interface (API) is included with PS/2 TV for organizations that need the capabilities of the PS/2 TV in a program-controlled interactive environment. The *PS/2 TV User's Guide* documents the API, describes how to code to it, and gives examples.

An organization may want to store video segments as objects in a remote videodisc library. By writing a custom program to the optional API, organizations can provide users with automated access to these seg-

ments from desktops. You can simply select a video segment from a menu; in turn, the customized environment would direct the video distribution system to search for an open video channel. After selecting an open channel, the customized environment would then tune the PS/2 TV unit to the chosen channel and begin video transmission.

An automated system also could be used to play Digital Video Interactive- (DVI-) compressed digital video stored on remote CD-ROM or hard disk servers, with "on the fly" conversion to analog signals for transmission.

### How F-Coupler Works

With the IBM F-Coupler, an organization can now assemble a complete network for all its video, audio, and data needs using a single installation of a shielded twisted-pair IBM Cabling System.

Operating in a 4 Mbits or 16 Mbits per second token-ring environment, approximately 500 MHz of free bandwidth is available for video transmissions in the 50 MHz to 550 MHz range. This free bandwidth, shown conceptually in Figure 4, can easily support over 70 channels.

Frequency Division Multiplexing (FDM) techniques make this greater

cable bandwidth utilization possible. Specifically, the ICS uses high-quality cable that can support signal transmissions at the high frequencies used for broadband audio and video, as well as the data transmissions for which it has been used historically. This shielded, twisted-pair cable contains both aluminum foil and mesh shield, enabling it to handle the simultaneous transmission of broadband and baseband signals.

With an IBM F-Coupler on each end of the cable, a higher frequency signal for audio and video is transmitted on top of the data signal. The head-end F-Coupler combines the two signals into a single transmission, while the receiving F-Coupler separates the signals and routes them to their appropriate ports. Each F-Coupler includes both a coaxial connector for broadband signals and a data connector for baseband signals. A circuit in the F-Coupler is designed to isolate the different signals from each other on their respective bandwidths, maintaining complete isolation during transmission.

### Flexibility of Use

The ICS/broadband LAN can be adapted into a variety of configurations, depending upon the unique needs of each user.

Video signals can originate from an antenna, VCR, or video camera. These broadband signals are then transmitted on coaxial cable to a multi-splitter (or tap/combiner) that leads to the CATV distribution panel in the wiring closet. It is here that the video signals are directed to the F-Coupler. Simultaneously, data signals also are transmitted on the IBM Token-Ring

network to the wiring closet, where they too are routed to the F-Coupler.

Leaving the wiring closet, shielded twisted-pair wiring on the ICS carries both baseband and broadband signals to the end of the cable drop. At that point, the F-Coupler at the faceplate separates the signals, sending the data to the PS/2 workstation while the broadband transmission is directed to a video receiver.

*Once cabled and configured, the IBM PS/2 TV unit is always ready.*

### Easy Installation

To install this system, a technician places one F-Coupler on the ICS distribution panel in the wiring closet where the video and data signals are received. Another F-Coupler is placed in the office or conference room where the signals are to be separated. A special faceplate is required to enable both coaxial cable connection of the video and a token-ring attachment.

Technicians installing the devices must consider the total length of cable in the network, as well as the length of cable in each node in the network. Recommended cable lengths can be obtained from the *F-Coupler Planning Guide* (GA27-3949) or other documentation such as the *IBM Cable System Planning and Installation Guide* (GA27-3361).

Similarly, installation of an ICS broadband LAN requires calculating the attenuation of the video signal as a function of both signal frequency and cable length. This information is available from the *F-Coupler Planning Guide*.

### References

In addition to basic product brochures, the following in-depth IBM information materials are available:

- *Video Monitoring Solution* videotape (GV21-8205)
- *F-Coupler Planning Guide* (GA27-3949)
- *F-Coupler Assembly and Installation Instructions* (GA27-3950)
- *PS/2 TV User's Guide* (G571-0238)

For more information about the IBM video monitoring solution, contact your IBM representative or call the IBM Multimedia Information Center at (800) 426-9402.

*Geoffrey R. Amthor is president of Impact Ideas, Inc., a consulting firm specializing in multimedia research and communications. Over the past seven years, he has published numerous articles in IBM publications as well as others. He has specialized in multimedia since 1989.*

*Impact Ideas, Inc.  
6713 Crestwood Peninsula  
Flowery Branch, GA 30542  
(404) 967-9700*



# Memory Address Space

**Rick Dayan**  
**IBM Corporation**  
**Boca Raton, Florida**

*This article discusses memory address space in several generations of microprocessors and how memory has been allocated in those processors. The concepts of system memory, nonsystem memory, and memory regions are introduced.*

Everyone is familiar with the famous 640 KB boundary for memory available for DOS applications. There is a similar situation in 80386 SX-based systems today. These systems can be populated with up to 16 MB of physical memory; however, only 15 MB of that memory can be accessed for system use. Although the 640 KB boundary and the 15 MB upper limit occur in computers that are several generations apart, the two situations are basically the same.

Why is it impossible to access some physical memory? What happens to that memory? How is it used? By answering these questions, this article can help users plan how much physical memory, and what kinds of memory, to add to their 80386 SX-based systems.

## What is Memory Address Space?

Every processor has *memory address space*, which is defined by the number of address pins on the processor chip plus one additional pin. The additional pin designates that the address bus contains a valid memory address.

A car analogy easily explains the difference between physical memory and memory address space. Suppose the seating capacity of the car is four. A four-seat car will work with just a

driver. It is still called a four-seater, regardless of whether it is fully occupied. You would not say that the car is a one-seater because only one seat is occupied.

The car's seating capacity is analogous to the computer processor's memory address space, and the car's driver is analogous to physical memory. By putting 2 MB of physical memory into a computer that has 16 MB of memory address space, the address space is still 16 MB, though only 2 MB are physically occupied.

Processors behave as though the entire memory address space always exists, whether or not the space is actually populated with memory. The user must decide whether to populate the memory address space with memory.

## System and Nonsystem Memory

Returning to the car analogy, suppose that the driver of the four-seat car goes grocery shopping. The groceries are placed in the two rear seats. Although the rear seats were intended for use by people, they can certainly be occupied by groceries. With the groceries there, the two rear seats are now unavailable for people to use.

In computers, memory address space is intended for use by system memory, but the space also can be popu-

lated by nonsystem memory. The nonsystem memory occupies memory address space that is ordinarily used for system memory.

What is the difference between system memory and nonsystem memory? *System memory* – which is owned, managed, and allocated by the operating system – is for general storage of operating system code, application code, and data. *Nonsystem memory* is not owned by the operating system; it has a dedicated purpose – typically used by feature adapters to provide programming interfaces, device control, and data buffers.

Memory address space can contain a mixture of several different memory technologies: Read-Only Memory (ROM), Random Access Memory (RAM), flash memory, Erasable Programmable Read-Only Memory (EPROM), and so on. Only RAM can be system memory. Nonsystem memory consists of memory other than RAM, as well as RAM memory that is dedicated to interfacing with a feature adapter.

In the car analogy, suppose that on the way home with the groceries, the driver spots two friends who need a ride. The driver stops to pick up the friends, but then realizes that there is only one unoccupied seat. The driver tells one friend to get in the car and tells the other to wait for another ride.

The computer similarity is this: Suppose that a computer system has 16 MB of memory address space, and that nonsystem memory occupies 2 MB of that space. The user has system memory modules that could fully populate the 16 MB space with system memory. However, the user finds it is impossible to add and access all the available system memory modules, because 2 MB of memory address

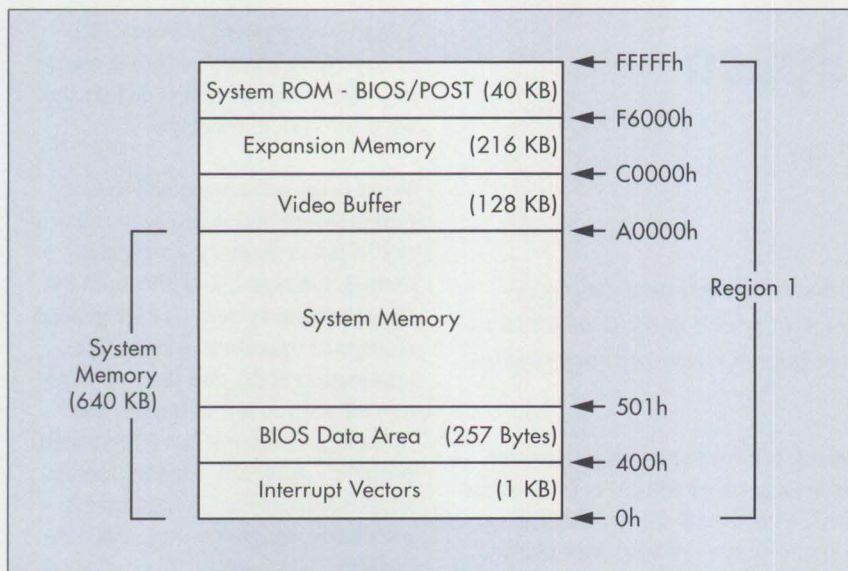


Figure 1. 1 MB Memory Address Space of the 8088 Processor

space are already occupied by non-system memory.

During configuration of the memory address space, allocation of nonsystem memory takes precedence over allocation of system memory. Nonsystem memory must take precedence because if a feature adapter requires memory as its interface to the system, then nonsystem memory is the only way to add feature-adapter functions to the system.

Allocations of nonsystem memory are not readily apparent to users. The existence of nonsystem memory becomes apparent only after system configuration, and only if the nonsystem memory replaces system memory or if the user views the system configuration using Setup.

### Address Lines

The total size of the memory address space depends on the number of address lines (also known as address pins or address signals) coming from the processor chip.

- The 8086 and 8088 processors have 20 address lines that support a 1 MB memory address space.

These processors can access addresses between 0 and FFFFh (where "h" denotes hexadecimal). FFFFh is the address 1 MB minus 1.

- The 80286 and 80386 SX processors have 24 address lines that support a 16 MB memory address space (from 0 to FFFFFFFh).
- The 80386 DX, 80486 SX, and 80486 DX processors have 32 address lines that support a 4 GB memory address space (from 0 to FFFFFFFFh).

This article discusses only the 8088, 80286, and 80386 DX processors. To apply the information in this article to other processors, refer to the processor relationships in the three bullets above.

From a programmer's viewpoint, there is a direct correlation between the number of address lines coming from the processor and the number of address bits used by programs to access data. For example, the 8088 processor has 20 lines of address; therefore, to access data with the 8088, the programmer uses a segment register plus an offset register.

The processor converts the contents of those registers to 20 address bits.

The use of memory address space has evolved with new generations of processors.

### 8088 Memory Address Space

The IBM Personal Computer and Personal Computer XT™ have a 1 MB memory address space in the 8088 processor, with 384 KB allocated as follows:

- 40 KB for I/O routines that mask I/O device idiosyncracies from the operating system and application software; contains Basic Input/Output System (BIOS) routines – the initialization vector (the first instruction fetched by the processor at power-on or reset), Cassette BASIC, and Power-On Self Test (POST)
- 128 KB for the video buffer
- 216 KB for expansion memory

Figure 1 shows the memory map detailing this information. Memory Region 1 represents the memory address space of the 8088 processor. Region 1, which ranges from address 0 to address 1 MB minus 1, also is called the *real mode* of the processor.

These dedicated functions in nonsystem memory are placed into the uppermost 384 KB of the 1 MB address space. Because it contains nonsystem memory, this area is not available to the operating system and applications for program and data storage. The lower boundary of this area is at 640 KB, or A0000h. The memory address space below 640 KB is for operating system code, application code, data, and input/output buffers. This is the origin of the 640 KB maximum for DOS programs. The 384 KB block of nonsystem memory became an important factor when memory increments increased in size and exceeded 640 KB.

Memory increments for the 8088 processor are either 16 KB or 64 KB. Those small increments meant that users could purchase the exact amounts of memory they needed, and the memory increments could fill any unoccupied gaps in the address space between the top of system memory and the 640 KB boundary. Therefore, all memory added to an 8088 system is utilized; there is no "extra" memory that cannot be used. (This contrasts with the situation in later generations of systems where minimum memory increments are 1 MB, and not all of that memory can be used by the system.)

The 640 KB boundary was not a problem in the early 1980s. Applications did not require much space and only one program resided in memory at a time. However, as users began to require more function and to run multiple programs simultaneously, the 640 KB boundary became a barrier.

The IBM PC/XT introduced the feature-adaptor expansion area that allowed feature adapters to obtain nonsystem memory allocations. The adapter expansion area was located above the 640 KB boundary at C0000h, so it made use of that previously reserved area. The adapter area did not affect the memory already assigned to operating system and application storage.

### 80286 and 80386 SX Memory Address Space

The next processor was the 80286 in the IBM Personal Computer AT®. From a memory address space perspective, the 80386 SX processor is identical to the 80286.

The 80286 processor increased the available memory address space from 1 MB to 16 MB. The 80286 had both the original real mode (Memory Region 1) and a new mode called *protected mode*. The 80286 had to run in protected mode to address the

region from 1 MB to 16 MB, which is Memory Region 2. Real mode (Memory Region 1) is the memory address space from 0 to 1 MB minus 1; protected mode in the 80286 processor is the memory address space from 0 to 16 MB minus 1 (not from 1 MB to 16 MB minus 1, which is Memory Region 2). The 80286 had to run in either real mode or protected mode; it could not combine the two modes. Therefore, protected mode had to cover the entire address space from 0 to 16 MB minus 1 (that is, both Memory Regions 1 and 2).

### *From a memory address space perspective, the 80386 SX processor is identical to the 80286.*

Because existing DOS applications did not use protected mode, the original DOS memory layout was superimposed on the memory address space of the 80286, and it became the standard for the real mode of the 80286. Although this enabled easy migration from 8088-based systems to 80286-based systems, the full capability of the new processor was not exploited until new software became available.

As shown in Figure 2, the 80286 generated new system requirements for using memory address space. The 80286 fetched its first instruction, the initialization vector, from a different address than in the 8088 processor. In the 8088 processor, the initialization vector was located at address FFFF0h in Memory Region 1 (the only region available). In the 80286, the initialization vector was located at address FFFFF0h in Memory Region 2. To provide an initialization vector, engineers had two alternatives. One was to modify the memory con-

troller to provide a second set of address decodes for the system ROM, which contained the initialization vector. (In effect, this created a logical *shadow* of the system ROM.) The second was to provide an additional ROM module with its own initialization vector at address FFFFF0h. IBM and other manufacturers chose the first alternative. This avoided adding a second physical ROM to the system, saving cost. At the same time, the size of the system ROM was increased to 64 KB because of new functions in the AT.

The result was that the second set of ROM address decodes required 64 KB in the 15th MB to be occupied. Just as 64 KB below 1 MB were taken from the operating system in the original PC, a second 64 KB, below 16 MB, were taken from the operating system in the AT.

To calculate the amount of memory address space used in the 80286 processor, add system usage plus areas reserved for feature adapter expansion and the video buffer. Figure 3 shows these calculations totaling 448 KB.

The maximum value of the 80286 memory address space is 16 MB. Subtracting the 448 KB from 16 MB leaves 15.5625 MB of available memory address space. Therefore, the most memory that users can place and enable in the memory address space for use by the operating system and applications is 15.5625 MB of RAM. But this is scarcely a limitation. It provided more capacity than most users needed, because DOS previously supported only 1 MB of memory address space. Memory expansion increments are small enough so that all unused gaps in the memory address space can be populated without having to disable any expansion memory.

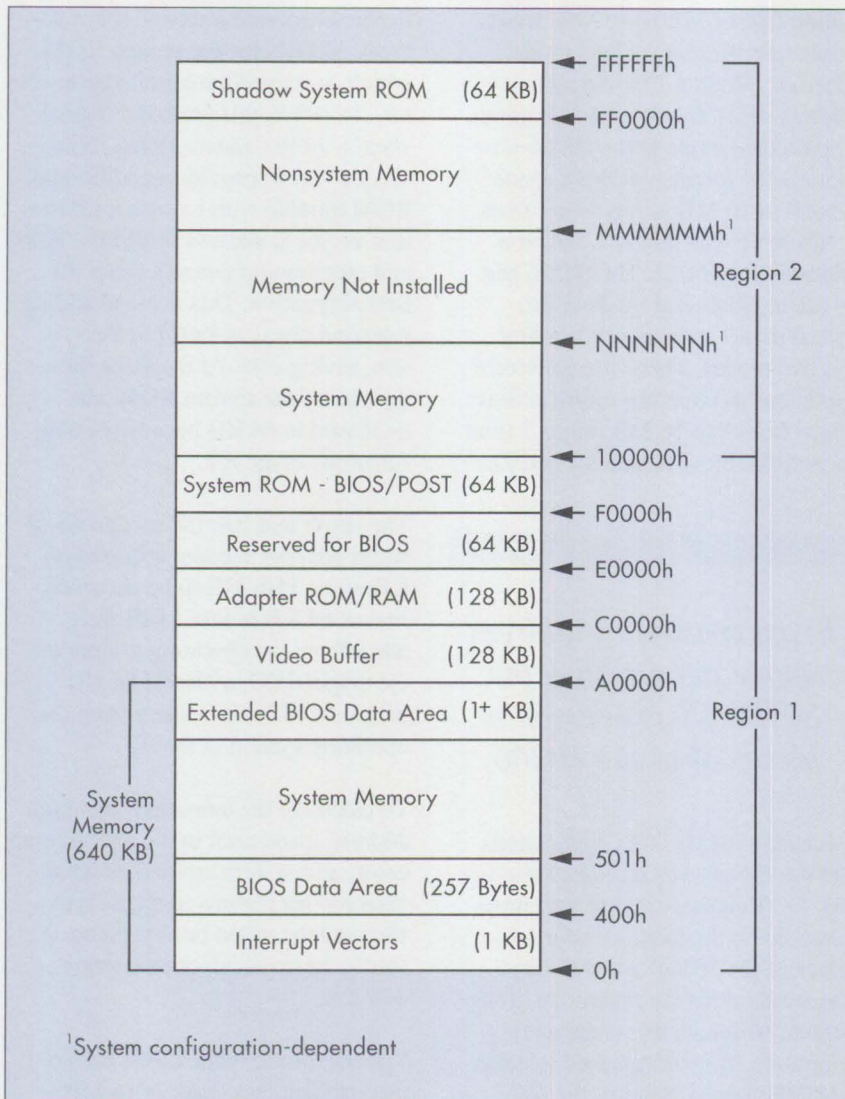


Figure 2. 16 MB Memory Address Space of the 80286 Processor

Item	Size	Region
Video Buffer	128 KB	1
Adapter ROM/RAM	128 KB	1
System ROM (BIOS/POST)	64 KB	1
Reserved	64 KB	1
Shadow System ROM	64 KB	2
TOTAL	448 KB	

Figure 3. PC AT Memory Dedicated or Reserved for System Functions

## 80386 DX Memory Address Space

In 1987, IBM introduced the PS/2. The PS/2 Model 80 contains the 80386 DX processor. In addition to the original 64 KB used for BIOS, PS/2 Micro Channel systems use the reserved 64 KB segment starting at E0000h (in real mode memory) for additional BIOS support. This brings the total BIOS usage to 128 KB of memory.

The 80386 DX processor has a third memory region. Memory Region 3 extends from 16 MB to 4 GB. The 80386 DX processor has the same two modes as the 80286: real and protected. As in the 80286 processor, the 80386 DX processor cannot be in both real and protected modes at the same time. Protected mode in the 80386 DX covers addresses up to 4 GB. Therefore, protected mode in the 80386 DX processor covers the entire memory address space, from 0 to 4 GB minus 1.

Because of advances in technology, additional memory comes in minimum increments of 1 MB. OS/2 and other new operating systems support more than 1 MB of memory. Adapters also use areas in memory above the 1 MB boundary for dedicated purposes, similar to the feature adapter expansion area in Memory Region 1.

All these new capabilities affect the memory address space map for the 80386 DX. With the introduction of Advanced BIOS (ABIOS), video on the system board, and extensions to POST, the 64 KB area of system ROM on the AT had to be expanded. The logical choice for expanding system ROM was the 64 KB reserved area in memory starting at E0000h, which immediately precedes the area occupied earlier by system ROM starting at F0000h. Introduction of Memory Region 3 caused the shadow of the system ROM to be moved from Memory Region 2 to the top

128 KB of Memory Region 3. Figure 4 shows the details.

Memory increments of 1 MB caused the physical memory in Region 1 to be split into a 640 KB and a 384 KB block. The 384 KB block is the *split memory block*.

Because of the limited area in Memory Region 1 for feature adapter expansion, adapters started using space in Memory Region 2. Using that space for adapters and devices results in reduced system memory available in Region 2. Under certain conditions, additional system memory may be lost, depending on the starting address for space dedicated to adapters and devices, and how much system memory is present.

For example, consider a PS/2 system populated with 16 MB of physical memory. Logically, this means that 15 MB of system memory is located in addresses from 0 to 640 KB and from 1 MB to 15.375 MB. Now suppose we want to add an Extended Graphics Array (XGA) display adapter to the computer. The installation program for the XGA adapter requests a 1 MB window starting at address 14 MB. But system memory is populated up to 15.375 MB, so 1.375 MB of system memory must be given up to provide space for the XGA window. Of the 1.375 MB, 1 MB must come from existing system memory, and the 0.375 MB is the 384 KB split memory block (0.375 MB = 384 KB).

The split memory block does not conflict with the XGA adapter memory. However, in each of the Memory Regions (1, 2, and 3), no system memory can be located above the start of nonsystem memory. Here, the XGA memory window is nonsystem memory, so the 0.375 MB split memory block also must be considered nonsystem memory.

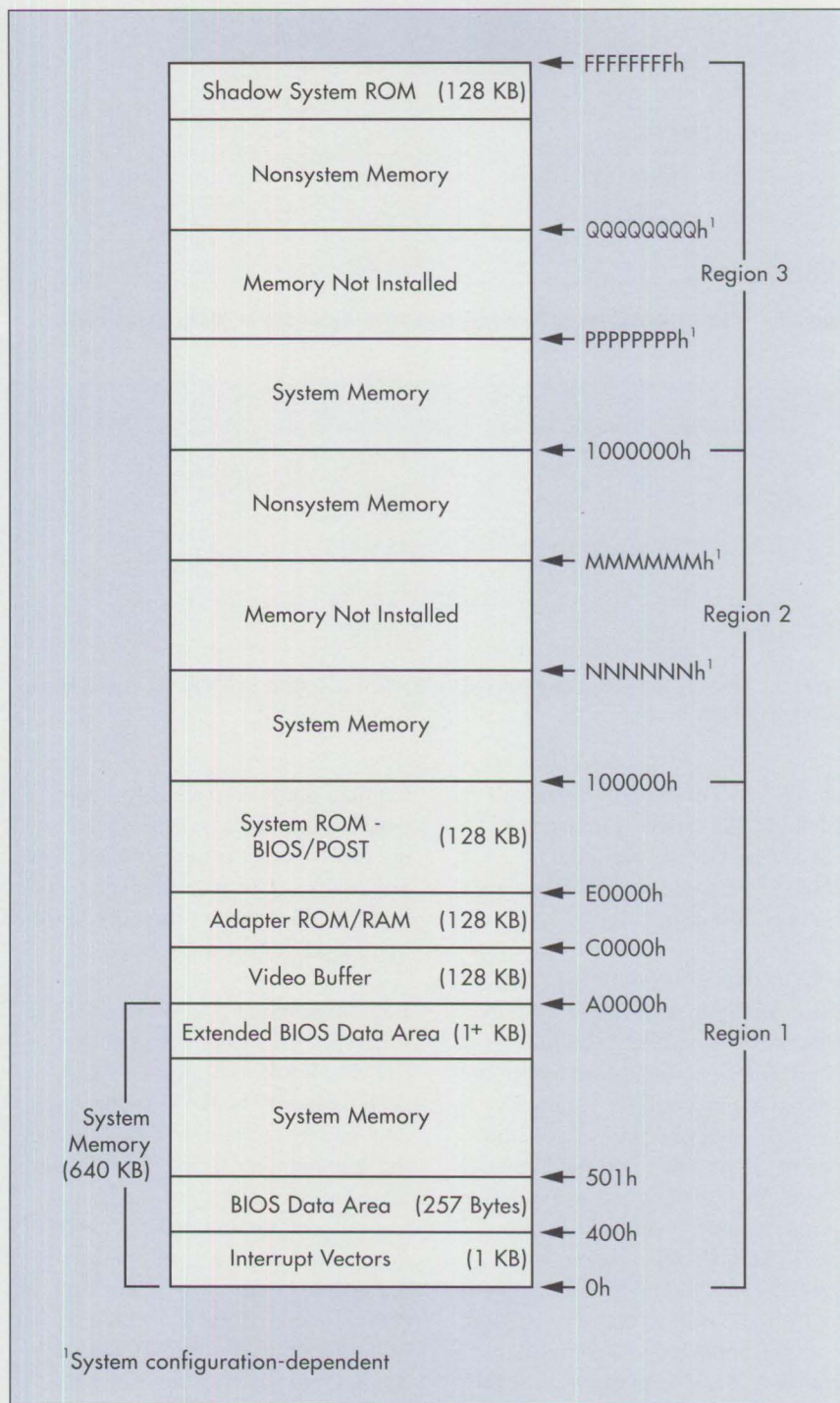


Figure 4. 4 GB Memory Address Space of the 80386 DX Processor

### Memory Utilization

Figure 5 shows the memory reserved for system usage in 80286-based Micro Channel systems; Figure 6 shows the memory reserved for sys-

tem usage in 80386 DX-based Micro Channel systems.

The memory utilization maps in Figures 5 and 6 can lead one to believe

Item	Size	Region
Video Buffer	128 KB	1
Adapter ROM/RAM	128 KB	1
System ROM (BIOS/POST)	128 KB	1
Shadow System ROM	128 KB	2
TOTAL	512 KB	

Figure 5. Memory Dedicated or Reserved for System Functions in 80286-Based PS/2s

Item	Size	Region
Video Buffer	128 KB	1
Adapter ROM/RAM	128 KB	1
System ROM (BIOS/POST)	128 KB	1
Shadow System ROM	128 KB	3
TOTAL	512 KB	

Figure 6. Memory Dedicated or Reserved for System Functions in 80386 DX-Based Micro Channel Systems

that 15.5 MB of memory are available in 80286-based systems, and that 3.9995 GB are available in 80386 DX-based systems. However, this is not the case.

In 80286-based systems, the 16th MB is totally lost for use as system memory because of the shadow system ROM. Specifically, an additional 896 KB are unavailable for use by system memory because expansion memory comes in 1 MB increments. This reduces the amount of available system memory from 15.5 MB to 14.625 MB. Under certain conditions, the split memory block remaining from the split in Memory Region 1 can be remapped into Memory Region 2. If a remapping occurs, this brings the system memory total to 15 MB. For example, PS/2 Models 80-041 and 80-071 remap the 384 KB split memory block from Memory Region 1 to Memory Region 2.

The 80386 DX-based systems, with 4 GB of memory, are similarly managed. Only a small percentage of

memory address space is lost compared to the vast available space, so the precise math is insignificant. This article does not concentrate on determining what kinds of memory to use in 80386 DX-based systems.

Beginning with PS/2 Model 80-111, PS/2 systems copied the system ROM to RAM to improve system performance. The RAM space was obtained by reclaiming 128 KB from the split memory block. This reduces total system memory by 0.125 MB, leaving a maximum of 14.875 MB of system memory enabled in the memory address space. This is also true of PS/2 Models 90 and 95, which have Initial Machine Load (IML) capability. In these systems, the 128 KB is used by the system to contain the IML image instead of the ROM BIOS image. However, the additional loss of the 128 KB in an 80486 memory space compared with the vast available space is so small that it does not warrant performing the calculations for the remaining maximum capacity.

## 80386 SX Memory Address Space

Next, the 80386 SX processor was introduced. Because the 80386 SX memory address space is identical to 80286 memory address space, all the previous information about 80286 memory address space applies to the 80386 SX.

The memory address space in an 80286-based system cannot be fully populated with system memory, because some memory address space must accommodate nonsystem memory for adapters and reserved areas for system use, such as system ROM. This also holds true for 80386 SX-based systems.

This is why the operating system and applications can access and use only 15 MB of the 16 MB memory address space in 80386 SX-based systems.

With this knowledge, users can decide what kinds of memory to use to populate the memory address space in an 80386 SX-based system. There can be a maximum of 15 MB of RAM system memory. The remaining memory address space is populated with nonsystem memory.

*Rick Dayan is a senior programmer in the IBM Entry Systems Technology laboratory in Boca Raton, Florida. After joining IBM in 1973, Rick has worked on the IBM 5230 Data Collection System, the IBM 5247 Disk Storage Unit, the development of the Personal Computer AT, and has been involved in many facets of the personal systems business. His current assignment is in Personal Systems Architecture. Rick holds BS and ME degrees in electrical engineering from the University of Florida.*

# OS/2 2.0 Installation and Performance Considerations

**Ron Morrill**  
IBM Corporation  
Roanoke, Texas

and  
**Ron Cadima**  
IBM Corporation  
Boca Raton, Florida

*This article and the article "OS/2 Application Support" in this issue discuss the improvements in OS/2 2.0 compared to OS/2 1.X; what they mean to users; and how to take advantage of them to tune systems for improved performance and usability<sup>1</sup>. Several major aspects of OS/2 2.0 are discussed, including installation, performance optimization, system internals, and application support. Installation subjects range from pre- through post-installation, and include LAN-based installations of OS/2 2.0. Performance optimizations discussed include improvements to the file systems, the lazy write capability, the flat memory model, and the paging system. OS/2 system resources and limits are listed in a chart that compares OS/2 1.3 and OS/2 2.0.*

OS/2 2.0 sets new standards for function, resource, and usability. By taking advantage of the Intel 80386 and 80486 processor architectures, OS/2 2.0 applications can gain significant performance improvements over earlier versions of OS/2. The Presentation Manager (PM), file systems, and other portions of OS/2 take advantage of the 32-bit addressing capabilities of the 386 and 486 hardware architectures. This, in turn, greatly increases the number of system resources – tasks, threads, files, and so on – that are available to programs running under OS/2 2.0.

## Pre-Installation Considerations

Before you decide how to set up an OS/2 2.0 system and how installation should proceed, read the README file that is shipped with the system. This file contains updated information that is not in the manuals, about subjects such as system tuning, functions and applications that do not work, and error recovery.

Before installing a system, you should know which applications will be running, their disk storage requirements, the amount of space required to save data, and your plans for future

growth. You also should know if the applications run on a LAN with all the programs and data stored on a server, or if the programs and data are stored on a local hard disk. Other factors in the operating environment include whether you run DOS and Windows applications; what kinds of OS/2, DOS, and Windows applications you run; when applications are run; and whether any applications run concurrently.

The Boot Manager, a major new feature in OS/2 2.0, enables you to have multiple operating systems installed in the computer, and allows you to select which operating system to boot. This is helpful when migrating from OS/2 1.3 to OS/2 2.0.

Assume that you are migrating disk partitions from OS/2 1.3 to OS/2 2.0. When you set up the hard disk with the Boot Manager, create the following in this sequence:

1. One partition for the OS/2 1.3 system and another partition for OS/2 2.0. The first partition should be allocated to the most-used operating system. Create all the partitions needed for operating systems before proceeding to the next step.
2. At least one partition for applications and data. Again, allocate the first of these partitions to the most-used programs and data, and create all the partitions needed for applications and data before proceeding to the next step.
3. The Boot Manager partition at the upper end of the hard disk. Locating the Boot Manager partition at

<sup>1</sup> If you have access to CompuServe<sup>®</sup>, IBM's National Solutions Center (NSC) BBS in Atlanta, or the OS/2 BBS, you should also refer to the OS/2 Tips and Techniques and the IBMOS2 forums that are available on them. These sources will give additional up-to-date information about OS/2 and its use. For more information about IBM's bulletin boards, contact your IBM representative.

the upper end reduces the seek time to all partitions that precede it.

It is best to have separate partitions for operating systems and for programs and data. This affords a level of security and data integrity because it reduces the chances that you will corrupt data when you apply operating system upgrades. Sometimes you may see improved performance because of the separate partitions. A third consideration is that if you are migrating from OS/2 1.3 to OS/2 2.0, and you decide that you are satisfied with OS/2 2.0 and no longer need OS/2 1.3, simply reformat the OS/2 1.3 partition and reclaim that disk space without concern about losing any data.

You also will want to use the Boot Manager if you install OS/2 2.0 on a computer that has DOS and Microsoft Windows 3.1 already installed. Although the dual boot facility of OS/2 1.3 is still supported within OS/2 2.0, there have been problems with using it after installing OS/2 2.0 on a computer that already had DOS 5.0 and Windows 3.1 installed. To avoid these problems, use the Boot Manager to install OS/2 2.0 in a partition other than the primary C: partition occupied by DOS and Windows, and ensure that the Windows .INI files for use under DOS are not the same .INI files used for WIN-OS/2 support under OS/2 2.0.

To use the Boot Manager on an existing hard disk, you need 1 MB of free disk space that is not allocated to any disk partitions. If 1 MB of free disk space is not available, you will have to reformat your disk to free up 1 MB of space. Remember that when a disk is reformatted, any partitions that are deleted and re-created lose all the data that was stored in them. Therefore, choose the partition with the least amount of data stored in it, and be sure to back up the data in the partition so it can be restored. Although the Boot Manager partition should be at the upper end of the hard disk, it can be placed anywhere on the disk. You probably will want to place the Boot Manager partition at a location other than the upper end, if the upper end is already occupied by a formatted partition containing data. It would be easier and more sensible to allocate the Boot Manager partition to available space elsewhere on the disk than to back up all data on the hard disk, reformat the entire hard disk, then restore all data to the disk.

The Boot Manager will support multiple physical disks, but it must reside on the disk from which you boot, and it must be marked as startable. You can create the Boot Manager partition only by using the FDISK.COM or FDISKPM.EXE programs that are shipped with OS/2 2.0. FDISK.COM

can be executed under OS/2 1.3 or 2.0, but not under DOS.

Whether or not you use the Boot Manager, you may want to consider creating a separate partition to contain updates to the operating system. Because the update procedure allows selected drives and directories to be maintained, you can create a test directory to apply the latest updates. Then, change the LIBPATH statement in CONFIG.SYS to point to this test directory first, and run OS/2 from the test directory for a time. If a problem arises during that time, it would be simple to change CONFIG.SYS back to the original LIBPATH and continue with your work. However, if all goes well, you can then apply the updates to the normal OS/2 directories. (For more details about LIBPATH settings, see the article "Cleaner Installation of Applications Under OS/2" in this issue.)

Using a small bootable OS/2 partition (about 3 MB of disk space) "off-loads" the normal OS/2 system so that you can update it. This circuitous route is necessary because it is not possible to update some OS/2 files when OS/2 is active. Booting from this small partition, however, allows updating any of the normal OS/2 files. If you also enable this small boot partition to establish a LAN session or a host asynchronous communication session, it may be possible to modify and maintain OS/2 from a central site.

Figure 1 can help you determine how much disk space is required for an OS/2 2.0 system partition.

There are several ways to reduce the amount of disk space needed by OS/2:

- Use selective installation to choose which OS/2 features will occupy hard disk space.

OS/2 Installation Option	Disk Space Required
Default Installation	27 MB
Preselected Installation	21 MB
Selective Installation with no options selected	13 MB

*Note:* These numbers cover only the OS/2 components. To these numbers, add at least 6 MB for the swap file and 5 MB for buffer space for new functions and maintenance fixes. If you install the Boot Manager, remember that it requires 1 MB of disk space. Also, if you intend to install Extended Services (ES) 1.0 or LAN Server (LS) 2.0, you must allocate sufficient space in the OS/2 partition for the files they will add. Refer to the ES and LS *Installation and Planning Guides* for more information.

**Figure 1. OS/2 2.0 Disk Storage Requirements**



- In a LAN environment, move some OS/2 files from the computer's hard disk to the LAN server.
- Install only the fonts that will be needed.
- Do not install all the available productivity programs and games.
- Do not select options (such as asynchronous communications) that you will not use.
- Allocate less disk space for the OS/2 swap file, because the more memory the computer has, the less disk space is needed for OS/2. The swap file acts as an extension of physical memory, so the more actual memory, the smaller the swap file can be.

During installation of OS/2 2.0, the system automatically sets the size of the swap file based on the amount of physical memory in the computer. Figure 2 shows the various space allocations for the swap file.

The initial size of the swap file is defined by the second parameter on

Real Memory Size	Swap File Size
4 MB	6 MB
5 MB	5 MB
6 MB	5 MB
7 MB	4 MB
8 MB	4 MB
9 MB	3 MB
10 MB	3 MB
11+ MB	2 MB

*Note:* When OS/2 boots, it tries to give its swap file the size specified in CONFIG.SYS. If there is not enough free disk space available, OS/2 starts with a 512 KB swap file, and dynamically increases the size of the swap file until it becomes large enough for OS/2 to initialize – that is, to be up and running.

**Figure 2. Swap File Space Allocation**

the SWAPPATH statement in the CONFIG.SYS file; the threshold for the swap file is defined by the first parameter. If the amount of free space left for the swap file is below the threshold, OS/2 warns that it is running out of space and suggests closing some active applications. The following line defines a swap file initialized to 2 MB with a threshold of 1 MB.

```
SWAPPATH=C:\OS2\SYSTEM 1024
2048
```

These numbers are for general use. If you always run specific applications and tasks, you should change the initial swap file setting to meet your actual usage. For example, if you look at the size of the swap file after completing a normal work day and see that it is 10 MB, change the SWAPPATH statement to the following:

```
SWAPPATH=C:\OS2\SYSTEM 1024
10240
```

Revise this setting if you change the amount of physical memory in the system, the size of the file system cache, or other system parameters that affect the amount of memory used by OS/2. The 10240 number used in the example represents the amount of memory allocated and used by OS/2 and the applications that could not fit in the system's memory (RAM). The 10240 can be reduced by the amount of physical memory added. If the size of disk

cache is made larger, then there is less memory for OS/2 and applications – therefore the size of the swap file must increase.

The 10240 number only defines the initial swap file size. It will grow to whatever size is needed, provided there is enough room in the disk partition. Growing disk files slows performance, so define the initial size of the swap file large enough so it will not have to grow in size as applications are run.

Figure 3 can help you determine the amount of RAM required by OS/2.

The numbers shown in Figure 3 assume that no paging (swapping) occurs. To these numbers, add the amount of memory needed for file system caches, Virtual Disks (VDISKS), additional device drivers and buffers, expanded memory, extended memory, and DOS Protect Mode Interface (DPMI) memory. Because it is assumed that no page swapping occurs, you must add the amount of memory required by the applications. The numbers specified for DOS compatibility and Windows compatibility are for single sessions. If you run multiple concurrent sessions, multiply those numbers by the number of concurrent sessions. Also, you must total the expanded, extended, and DPMI memory for each of those sessions. The performance buffer is extra memory for loading applica-

Function	Memory
Base code	2.5 MB
DOS compatibility	0.6 MB
Windows compatibility	1.0 MB
High Performance File System (HPFS)	0.5 MB
Spooling (while printing)	0.5 MB
Performance buffer	0.5 MB

**Figure 3. OS/2 Memory Requirements**

tions and switching screens. There will be little or no noticeable performance degradation when memory is overcommitted (that is, when the operating system and applications require more physical memory than the computer has available) by only 5% to 10%. These numbers are not guaranteed to be valid in all user environments.

## Installation Process

There are several options for installing OS/2 2.0. You can install it from diskettes using the menu-driven install procedure or through a LAN. The server can be an OS/2 LAN server, a Novell server, or any other server system that supports TCP/IP. For detailed information about available installation functions, and about setting up for LAN installation of OS/2 2.0, refer to *OS/2 2.0 Remote Installation and Maintenance* (GG24-3780) and the *LAN Installation Utility/2 (LIU/2) User Guide and Reference* (SNMS-0001-00).

Using a response file can make installation easier. The response file simulates the responses given by a human operator to installation questions and options. The file RSPINST.EXE in the \OS2\INSTALL subdirectory allows you to install OS/2 2.0 on a LAN using a predefined response file as input. Its input is the name of a response file. The file SAMPLE.RSP, also in the \OS2\INSTALL subdirectory, contains information about the different options available. When specifying these options, a zero represents the default value (no option is selected), and a 1 selects a specific option or all available options. For example, to choose which system utilities to install, 0 means install none, 1 means install all, and a third option is to select the number of each individual utility to install. When multiple options are selected, they should be separated by a comma.

Following are some of the more important available installation options:

- **AlternateAdapter.** Specifies a secondary display adapter.
- **BaseFileSystem.** Specifies the file system that will be used to format the root directory if the Format Partition option is chosen. The 1 is for High Performance File System (HPFS) and 2 is for File Allocation Table (FAT).
- **DefaultPrinter.** Specifies the default printer to be installed for the system. The first printer diskette shipped with OS/2 2.0 contains a file called PRDESC.LST. Browse this file with a program that displays line numbers, find the line number of the printer you want, and use that number as the default in the DefaultPrinter statement. For example, DefaultPrinter=5 will install the Apple® LaserWriter® II NT printer driver. *OS/2 2.0 Remote Installation and Maintenance* has a table of the printer drivers and their index numbers. If printers are added to or deleted from the list, use the line numbers in the PRDESC.LST file.
- **DOSEnvironment.** Specify this option if DOS or Windows applications will be run.
- **WindowedWIN-OS/2.** Specify this option to run Windows applications from the OS/2 desktop in a seamless manner. If it is not selected, you can run Windows applications only from a full-screen environment.
- **WIN-OS/2Desktop.** Specifies whether you want a new WIN-OS/2 desktop or one that is copied from an existing Windows desktop. Together with this option, you can use the ExistingWindowsPath and ShareDesktopConfigFiles options to directly link the WIN-OS/2 desktop and the Windows desktop. *Important:* These options should be used only if you have Windows 3.0 installed on the computer. *Do not use these options with Windows 3.1, because they will corrupt either the Windows 3.1 or the OS/2 2.0 installation.*
- **DPMI.** Select this option if WIN-OS/2 support is installed or for any DOS applications that use DPMI. DPMI is a protocol for how DOS applications access and use memory above 1 MB in a protected environment.
- **EarlyUserExit.** Specify this option with the name of a command file or executable program to perform initialization steps before the installation starts. Use this option to partition the disk where OS/2 2.0 will be installed or to save some files before the installation proceeds. *OS/2 2.0 Remote Installation and Maintenance* includes an example of a REXX command file that partitions the disk.
- **UserExit.** This option is similar to the previous one except that it is processed after OS/2 2.0 installation is complete. It can be used to install additional printer drivers or applications or to restore saved files. *OS/2 2.0 Remote Installation and Maintenance* contains a sample REXX command file for installing additional printer drivers.

There are additional options for modifying the CONFIG.SYS file; selecting fonts, tools, and games; installing device drivers; and other system options. Refer to the SAMPLE.RSP file for further information.

When you install OS/2 2.0 by responding manually to the installation prompts, OS/2 creates a file that contains your responses. This file called USER.RSP is in the \OS2\INSTALL subdirectory. It can be used as a response file during subsequent installations of OS/2 2.0 on

other computers that have the same hardware configuration.

The OS/2 2.0 installation process automatically installs HPFS support if there is an existing HPFS partition on the disk. If there are no HPFS partitions on the disk and you have not done a selective installation of HPFS, then HPFS will not be installed on computers with less than 6 MB of memory, due to the memory requirements of HPFS.

The installation process also presets the disk cache size, based on the amount of physical memory in the computer and how much disk space is allocated to each file system. Figure 4 shows the various disk cache size allocations provided by the installation process.

These cache settings should be tuned to the computer and operating environment. Disk-intensive applications, such as file servers and database servers, should have larger caches; whereas computers that perform much data processing or those with a limited amount of RAM memory should have smaller caches.

## Post-Installation Considerations

After installation, much can be modified and tuned to make OS/2 more efficient and productive. Most are changes to the CONFIG.SYS file. Following are several CONFIG.SYS options that can noticeably affect the performance of OS/2.

### BUFFERS

The default for BUFFERS is 30. Because of changes made to the file systems, this parameter does not have as great an effect on performance as in previous versions of OS/2. The only time to set this parameter higher is when OS/2 is performing direct disk Input/Output (I/O), bypassing the file system caches. Because each buffer

occupies about 500 bytes of system RAM, it is possible to free up some memory. For a FAT file system, BUFFERS can be reduced to 10, saving about five pages of fixed memory (memory that cannot be swapped to disk and used for another purpose). For an OS/2 installation that uses only HPFS, or for DOS or Windows applications that are disk-intensive, BUFFERS can be reduced to 3. If you use file write-through, it may be beneficial to set BUFFERS equal to the number of blocks per track on the disk. Several DOS programs can be used to determine this number. One example is the SMARTDRV.SYS driver, which displays the number of blocks per track when booting a DOS system. Usually, cache space is used more efficiently than buffer memory and will give better performance.

### MAXWAIT

The MAXWAIT parameter should be changed only if the same applications are always run the same way. It is meant to give a low-priority task more of a chance to run. To improve the priority of a task, reduce the value of MAXWAIT. However, be careful about lowering this value, because important tasks may get less

processing time, adversely affecting the overall performance of the system. Therefore, instead of changing the MAXWAIT parameter, change the operating environment based on which applications you run, when you run them, and the order in which they run. If MAXWAIT is changed, it should be done in conjunction with the TIMESLICE parameter that can be added to the CONFIG.SYS file. With MAXWAIT=1 and TIMESLICE=50,75 (where the two parameters on the TIMESLICE statement represent minimum and maximum time slices in milliseconds), applications visible on the screen will appear to run more smoothly and consistently. However, you may pay a penalty in total throughput. These parameters should be changed to meet individual processing requirements.

You also should add the TIMESLICE parameter to CONFIG.SYS if certain types of DOS applications are run. Examples are programs that do full-screen graphics, such as games, and programs with a large amount of I/O polling, such as some communication packages. For these types of applications, also reduce the Idle Sensitivity setting in the DOS session that is set

System Memory	Cache Sizes When Both HPFS and FAT are Used	Cache Sizes When Either HPFS or FAT is Used
4 MB	128 KB and 64 KB <sup>1</sup>	128 KB
5 MB	128 KB and 64 KB	128 KB
6 MB	256 KB and 64 KB	256 KB
7 MB	256 KB and 128 KB	256 KB
8 MB	256 KB and 256 KB	384 KB
9 MB	256 KB and 256 KB	384 KB
10 to 16 MB	512 KB and 512 KB	1 MB
17 to 32 MB	1 MB and 1 MB	2 MB

<sup>1</sup> When both HPFS and FAT are used, the larger cache size is assigned to the file system that has the most disk space. For example, if the hard disk is 70% allocated to HPFS and 30% to FAT, and the computer has 6 MB of memory, the HPFS cache is initialized to 256 KB and the FAT cache to 64 KB.

Figure 4. Disk Cache Size Allocations

up to run the application. Start with `TIMESLICE=64, 128`, then adjust the settings from there. Usually Idle Sensitivity and `TIMESLICE` do not affect total system throughput, and they can improve the responsiveness of interactive systems. This change should be done only if unacceptable system performance is encountered.

### THREADS

This parameter controls the number of threads that can be used in OS/2. The default for OS/2 2.0 is 256. If the system is memory-constrained, this value should be calculated and changed. Most Presentation Manager applications use two or three threads each, so a conservative calculation formula is as follows:

$$\text{threads} = (N \times 3) + 60$$

*N* is the number of DOS + Windows + OS/2 applications. If the calculation is less than 128 threads, use 128.

If the system is running OS/2 Extended Services, OS/2 LAN Server, or both, you should add 128 threads. Also, if you know that an application uses more than three threads, add the extra threads to your calculation.

### MEMMAN

`MEMMAN` has several purposes. First, its `PROTECT` parameter specifies that OS/2 will allow protected Dynamic Link Libraries (DLLs) sometimes called Dynalinks – dynamic linking of data and programs in OS/2 – and will allow certain Application Programming Interfaces (APIs) to allocate protected memory. (The opposite is `NOPROTECT`.) With the `SWAP` (or `NOSWAP`) parameter, you can specify whether swapping is to occur; with the `MOVE` (or `NOMOVE`) parameter, you can specify whether memory movement is allowed. Memory movement means moving data from one place in memory to another. Movement is not allowed if an application accesses

that data using a specific memory address.

The defaults for `MEMMAN` are `PROTECT`, `SWAP`, and `MOVE`, and should normally remain that way. Change these parameters only if you run applications that you have developed, and if the machine has enough physical memory to accommodate OS/2 and the applications.

## *Most Presentation Manager applications use two or three threads each.*

Another parameter of the `MEMMAN` statement is `NOPACK`. All 16-bit applications are based on a segmented memory model. In OS/2, each segment of 4 KB or less is mapped to 4 KB pages. For example, a 200-byte segment is mapped into one 4 KB page. Because this is very inefficient and can cause significant growth in the swap file, OS/2 2.0 packs multiple small 16-bit segments into a single 4 KB page. Depending on the application, this can significantly reduce the size of the swap file. However, with some applications, a very small performance degradation can be experienced. For those cases where this presents a larger problem than increased disk usage, you can specify the `NOPACK` option in the `MEMMAN` statement. `NOPACK` negates packing multiple small 16-bit segments together and causes each segment, regardless of its size, to be mapped to a 4 KB page.

### PRIORITY

The default setting for the `PRIORITY` option is `DYNAMIC`. This setting should be changed only on a specific,

dedicated system. The option `ABSOLUTE` means that the system will not change the priority of regular tasks in the system based on screen focus (the window on the display screen that you are currently using with the keyboard or a mouse), I/O activity, and frequency of processor use. Changing this setting to `ABSOLUTE` enables you to predict when tasks will run, but it can cause the system to appear sluggish and unresponsive. Before changing the setting of `PRIORITY`, know the priorities of all active tasks and be able to set them to meet your needs.

### SWAPPATH

The swap path should point to the most used partition on the least used physical drive. To enhance performance, place the swap file in the same directory as work files and temporary files that are used often and change size frequently. This will help reduce fragmentation problems and increase the efficiency of disk accesses. Set the initial size of the swap file to the average working size in the environment. (*Working size* is the amount of memory and disk space required to run all programs and tasks that are normally done during one work day.) You also should set the swap threshold at 2 MB. This threshold will give you a minimum of four warnings before the system runs out of swap file space and you have to shut it down.

### PRIORITY\_DISK\_IO

The default for `PRIORITY_DISK_IO` is `ON`. This means that the application running in the foreground – the one that the user is interacting with – gets a higher disk I/O priority than applications running in the background. This action improves the response time of the foreground application. Setting `PRIORITY_DISK_IO=OFF` eliminates the higher disk I/O priority for the foreground application.

### Path Statements

The PATH, LIBPATH, and DPATH statements should be organized so that the most frequently used files are listed first and the least frequently used files are listed last. This sequence decreases the amount of searching, thereby improving performance.

*Note:* The OS/2 2.0 installation process places . ; at the beginning of the LIBPATH statement. This tells OS/2 to start its search for DLLs, fonts, and drivers in the currently active directory. Some applications modify the LIBPATH statement by placing their DLL subdirectory at the beginning of the LIBPATH statement. This can cause errors if more than one DLL file has the same name. After you install any application, including LAN Server 2.0, check the LIBPATH statement in CONFIG.SYS to ensure that it is correct. For more details, refer to the article "Cleaner Installation of Applications Under OS/2 2.0" in this issue.

### BASEDEV=OS2SCSI.DMD

This statement will appear in CONFIG.SYS if OS/2 is installed on a computer that supports Small Computer Systems Interface (SCSI) devices. This statement is required only if the system has SCSI tape drives, CD-ROM drives, or SCSI devices other than disks and diskettes.

### BASEDEV=IBM2SCSI.ADD

This statement in CONFIG.SYS defines the SCSI disk device driver. The device driver implements the System Control Block (SCB) architecture interface. It allows requests from the file system to be chained together in a list, so that the file system does not need to interact with the device driver after every request. For example, the file system can pass a request list to perform a seek, a read, another seek, and another read operation. When the device driver completes all four tasks, it passes control



back to the file system. The number of interactions between the file system and the device driver is greatly reduced, and performance increases. However, in cases where immediate response is required – such as searching a database for credit authorization – it is not possible to wait for multiple disk operations to be performed before getting a response. Use the /GS:n parameter to specify how many requests the file system sends to the device driver in a single SCB list. The value of n can be between 1 and 9; the default is 4. Consider increasing this number if the system is running applications that use work files heavily.

### SET RESTARTOBJECTS=ON

This undocumented capability, which is subject to change in a future release of OS/2, can be added to the CONFIG.SYS file. It determines the automatic startup of applications. The default is ON, so that any applications that were active when OS/2 was shut down are to be restarted the next time it comes up. If you specify OFF, no applications will be restarted. You also can specify the value STARTUPFOLDERONLY, which will restart only the applications in the Start folder.

### WIN-OS2 Considerations

The "OS/2 2.0 Application Support" article in this issue goes into detail about DOS and Windows support, and all of the DOS settings that can be used to customize and optimize DOS and Windows sessions. Here are a few things that affect system performance with respect to DOS and Windows support. Many DOS and Windows applications perform *device polling* to see if they have any work waiting for them. The most common use is when applications are waiting for input from the keyboard. Applications enter into a software loop asking the keyboard device if the user has pressed a key. Under DOS and Windows, this does not cause a problem because only one application is running at a time. With OS/2, there may be multiple programs running concurrently, which can cause other programs to slow down. To alleviate this problem, here are some things that can be done.

1. Reduce the Idle Sensitivity entry of the DOS Session Settings to 30 or less. This is a threshold used by OS/2 to determine that an application is polling a device and has no other work to do. By reducing this threshold, more time is given to other applications that have real

work to do. Change this setting whenever there are DOS or Windows applications performing graphics, asynchronous communications, keyboard polling, or other types of device polling while running in the background. You must set this for each one of those sessions using the DOS Properties menu of the Sessions Settings.

2. Memory utilization is a major factor when running DOS and Windows applications. Running DOS and Windows applications under OS/2 is equivalent to running multiple DOS machines. For example, if you have a DOS computer using 1 MB of memory, and another using 2 MB, when you bring those together in an OS/2 system they will require 3 MB total. This tends to make OS/2 swap RAM to disk. The effects of this can be lessened by adding a `TIMESLICE` parameter to the `CONFIG.SYS` file. In a memory-overcommitted system, most of the application's time slice is spent paging-in the memory which the application needs. It is beneficial to give the application more time to do its work. Try setting the `TIMESLICE` parameter as follows:

```
TIMESLICE=64,128
```

This says that the program will receive a minimum time slice of 64 milliseconds and a maximum of 128 milliseconds. Depending on your applications, you may want to change either the minimum or maximum values to meet your requirements.

Another factor to consider about memory is the amount of extended memory – DPMM, XMS, or EMS – that Windows applications use. If you are going to run multiple Windows applications in one WIN-OS2 session, you must ensure that the DOS settings for that session provide the total amount of extended memory required by all

the applications. If three Windows applications are run concurrently under the same WIN-OS2 session, and each requires 2 MB of DPMM memory, then you must allocate 6 MB of DPMM memory to the WIN-OS2 session. If the applications are running in separate sessions, then each session must define 2 MB of DPMM memory.

Refer to the DOS/Windows section in the “OS/2 2.0 Application Support” article in this issue for additional performance information and for details about all the DOS settings and how to set them.

*Many files can be moved from OS/2 clients to the LAN Server, where those files can then be used by all clients attached to the server.*

### Printing

Printing is a major concern in many system installations. This section describes some factors that affect printer support and performance.

#### Print Servers

Make sure that the same printer driver levels are on both the server and the workstations attached to the server. Print servers must run OS/2 1.3 or later. If they are not running OS/2 2.0, use the printer drivers that are shipped with OS/2 2.0. If the printer drivers on the workstation and the server do not match, performance will be significantly reduced.

#### Printer Data Streams

If your printer supports fonts, ensure that the correct fonts are installed in the printer. Whenever possible, send raw data streams of commands and data to the printer. This allows a job

to print while it is being spooled; otherwise, all the print data must be placed into the spool file before the printer can begin. Also, if you have a choice of printers, keep in mind that OS/2 2.0 can download fonts to a printer. OS/2 is aware of the printer and the fonts installed in that printer; if fonts are installed, OS/2 does not waste time downloading them. The separator page capability is new in OS/2 2.0. Separator pages can be specified in the Print Options page of the Printer Object settings. Two sample separator page files, `PSCRIPT.SEP` and `SAMPLE.SEP`, are in the `\OS2` subdirectory.

#### WIN-OS/2 Printing

Windows printer drivers must be installed separately from OS/2 printer drivers. These drivers are on the third printer diskette shipped with OS/2 2.0. For drivers not shipped with OS/2, install the drivers that are shipped with Windows. You also can use the printer drivers shipped with Windows 3.1 and see some improvement in performance of WIN-OS/2 printing, as you would in a Windows 3.1 environment. *Note:* This has worked in lab tests, but there is no guarantee that it will always work correctly.

For further information about printing, refer to *OS/2 Version 2.0, Volume 5: Print Subsystem* (GG24-3775).

#### LAN Requester Considerations

Many files can be moved from OS/2 clients to the LAN Server, where those files can then be used by all clients attached to the server. This significantly reduces hard-disk storage requirements of the client computers. It also provides an efficient way to limit a client's access to certain features and functions. For example, you can limit which clients can run DOS or Windows programs, which clients can run certain OS/2 utilities such as `FORMAT` and `FDISK`, and which clients can access third-party or in-house applications.

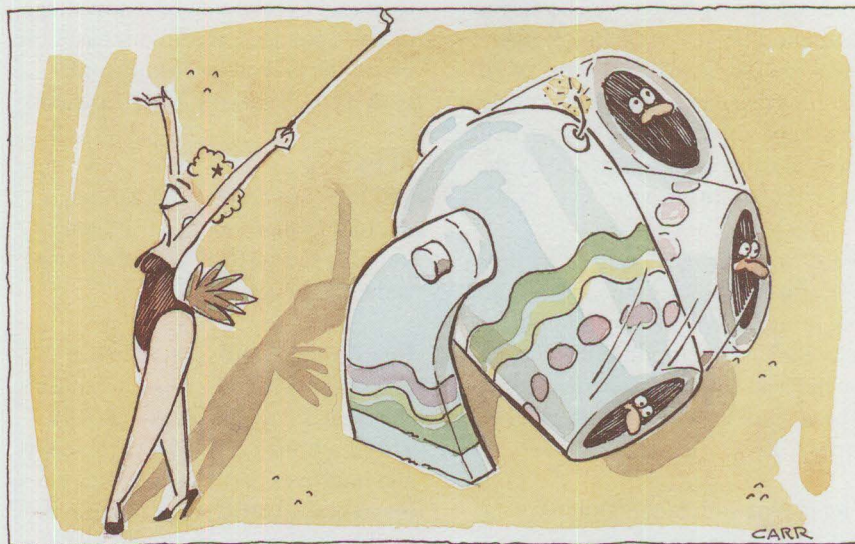
The following changes make it possible to produce a client system that supports LAN and 3270 emulation, DOS and Windows, and all OS/2 applications, while requiring less than 30 MB of disk space (excluding the SWAPPER.DAT file). The following changes are grouped according to the amount of work that needs to be done.

The easiest is to simply move files from client computers to the server computer, and to ensure that the path statements are modified to point to files on the server. The next level of work encompasses the first level, and also includes modifying the settings in all OS/2 folders that are affected. The last step requires running various tests to determine which files are not used, then moving them to the server.

A general rule is to keep all files that have extensions of .INI, .LOG, .MSG, .SYS, .PRO, and .DCP on the client computers. You should not delete any subdirectories, even if they are empty, because certain programs rely on having a subdirectory in a certain path. For all the subdirectories that have moved to the server, move their file names to the ends of the PATH statements. Ensure that any file moved to a server will not be used before the client computer logs onto the server.

There are several ways to handle the startup of clients. The easiest is to create a STARTUP.COM file that starts the requester and performs the logon function. Anything else that you would normally have in the STARTUP.COM file should be moved to another command file. In this new .CMD file, there should be a little program that waits for the logon to complete before proceeding.

Although not recommended, another alternative is to create a .CMD file and place it in the SET RUNWORKPLACE= statement in the CONFIG.SYS state-



ment in place of PMSHELL.EXE. This command file starts the requester and logs it onto the server, then starts the Workplace Shell™ by executing PMSHELL.EXE. This ensures that the client is logged onto the server before the Workplace Shell is started. For example, you could use the following statements in a command file called LOGON.COM:

```
NET START REQ  REM Starts
                requester code

LOGON USERID   REM USERID is
                the logon
                user ID

PMSHELL.EXE    REM Starts
                the Workplace
                Shell
```

Next, modify CONFIG.SYS as follows:

1. Add the statement  
SET SETRESTARTOBJECTS=OFF.
2. In the statement  
SET RUNWORKPLACE=, change  
C:\OS2\PMSHELL.EXE to  
C:\START.COM (assuming OS/2 is  
installed on the C: drive).

There are several items to notice when doing this. First, there is an extra CMD.EXE session running in the system. It uses extra resources and as much as 150 KB of memory. Second, if the Workplace Shell inadvertently

stops running, it will not automatically restart; the system has to be rebooted to bring it back. When you issue the Shutdown command, you may never see the "Shutdown Complete" message. For these reasons, and because SET RESTARTOBJECTS is an undocumented function, this method is not recommended.

### Simple File Moves

Because many OS/2 files are rarely accessed, they can be moved from client to server. If the server is running OS/2 2.0, most of these files should already exist on the server. Move all files with extensions of .LIB, .H, and .INC to the server. These files are usually required on the client computer only if applications are developed on it. Even then, the PATH statements for performing the compilations and linkages can be modified to point to directories on the server.

Executable files that are not defined in any OS/2 folders can be moved to the server. These files include all utilities in the \MDOS subdirectory and many utilities in the \OS2 subdirectory. All Help (.HLP) files and all Information (.IPF) files can be moved to the server. When this is done, the SET BOOKSHELF entry must be updated in CONFIG.SYS to point to the

relevant server directory. The current setting of the SET BOOKSHELF statement in CONFIG.SYS has a list of all the files that can be moved. You also should be able to move the files that are in the directory pointed to by the SET GLOSSARY entry.

*Note:* Some .LIB files, used by Easy-View to define screen images, should not be moved. Some applications may look for their .HLP files based on the installation or root directory. These .HLP files also should not be removed.

### File Moves with Modifications

All utility files that are defined in folders can be moved from a client to the server, but you must go to the specific folders and modify the settings for those file definitions to point to the server directories. You also should ensure that any SET or PATH statements in CONFIG.SYS are updated to point to the new server directories. Some files, such as ATTRIB.EXE, COMMAND.COM, and APPEND.COM, are required to start DOS sessions, so these files cannot be moved to the server.

### "Brave Soul" Updates

This applies to users who need every bit of space possible on the hard disk of the client computer. Make all the changes discussed above, then start OS/2 and all the programs that will be run. Next, make backup copies (on diskettes) of all the files in the LIBPATH, then delete all files with extension .DLL. All the .DLL files that are successfully deleted from the client computer can then be copied from the backup diskettes to the server. These .DLL files that are not deleted will remain where they are. Be sure to place the server directory path name in the LIBPATH statement in the CONFIG.SYS file.

### File Systems

This section covers the changes that have been made to the OS/2 file sys-

tems and tells how to take advantage of these changes. It also discusses the use of the lazy write feature under both the HPFS and FAT file systems.

### HPFS

OS/2 2.0 has several significant internal changes to the HPFS that improve the performance of disk I/O and OS/2 in general.

**Read Ahead:** HPFS provides an asynchronous read-ahead thread for files that are accessed sequentially. This loads the cache asynchronously, and improves performance because the next sequential record required by the application is already in the cache before the application issues the read request.

Read-ahead is done only for records of 8 KB or less. Larger records cause the HPFS cache to fill up with the records from only one file, and I/O performance degrades for all other files. Records larger than 8 KB are still placed into the cache if they are smaller than the cache threshold.

### Disk Device-Driver Strategy Protocol:

The SCB protocol for support of SCSI devices is new in OS/2 2.0. This allows command chains to be sent to the disk device driver and supports the scatter/gather capabilities of Direct Memory Access (DMA) bus-master devices. There is emulation for this protocol for Enhanced Small Device Interface (ESDI) devices. The file system looks at the type of device being used. If it is an ESDI device, it will try to obtain contiguous memory for the operation and use a type of STRAT 1 interface to the device. (If contiguous memory cannot be obtained, the normal emulation path will be performed.) In a STRAT 1 interface, all data being written to a disk must reside in a contiguous memory block. Data is transferred from this block in sections of 64 KB or less, because that is the maximum size of DMA requests that

ESDI devices can handle. Each transfer must be requested separately; transfers cannot be chained together in a request list.

**HPFS Cache Threshold:** The cache threshold parameter for HPFS is also new in OS/2 2.0. This parameter, in the IFS statement in CONFIG.SYS, specifies the size of the largest record that will be cached. The default is 4, which denotes 4 KB. Care should be used when setting this parameter. It should be set equal to the size of the record that is used most often. If the system primarily accesses two files, one with 8 KB records and one with 4 KB records, set the cache threshold value to 8. If another file is accessed with 64 KB records, increase the cache threshold only if that file is accessed more frequently than the other files, and if individual records are accessed more than once. If this is not the case, keep the cache threshold low so that the more frequently used records do not get pushed out of the cache.

The size of the HPFS cache should be set based on the amount of physical memory in the computer, the types and frequencies of disk I/O, and the application mix that will be running. HPFS requires more memory to support it than the FAT system requires. You must use a different approach to setting the cache size in HPFS than for FAT. HPFS requires approximately 350 KB of memory. OS/2 2.0 initializes the HPFS cache based on the amount of available memory. If you plan to run DOS/Windows applications or large OS/2 applications, it may be more beneficial to reduce the cache size and to provide more real memory for these applications. However, if you run smaller applications or applications that primarily perform disk I/O, it may be more beneficial to allocate a larger cache size. If a hard disk or partition has less than 60 MB, you may choose not to use HPFS because it would



leave less memory for files, negating any performance gains over a FAT system.

If the amount of memory required for HPFS is too much for the machine, or if you choose not to use HPFS because disk partitions are small, use the Selective Install option when installing OS/2 to deselect the HPFS option.

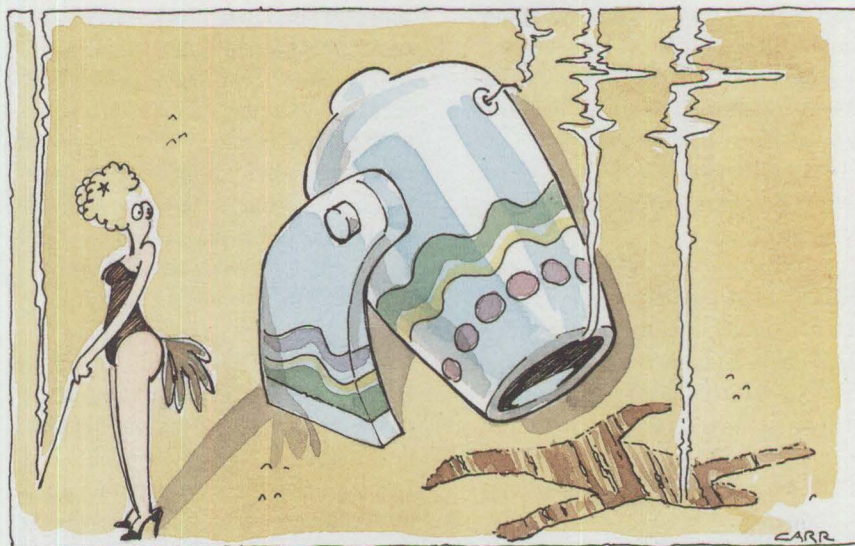
If the size of the HPFS cache or the cache threshold must be changed in CONFIG.SYS, do it after OS/2 installation is complete.

Other changes have been made to HPFS to improve disk I/O operations, to improve DosQueryPathInfo processing, and to improve cache processing. HPFS supports directories up to 512 GB in size and file sizes greater than 2 GB. HPFS does not yet support files that span multiple hard disks.

## FAT

OS/2 2.0 has several significant changes to the FAT file system. The most significant are the lazy write and sequential read-ahead capabilities found in HPFS. Lazy writing is covered in the next subsection because it is basically the same for both HPFS and FAT. The sequential read-ahead capability is the same as in HPFS, except that 16 KB read-aheads are performed for FAT. The use of different strategies for interfacing to disk device drivers is the same as in HPFS. From an application point of view, no changes have been made to the APIs that are used to interface to the file system, except that now the write-through parameter has meaning for the DosOpen API – it allows an application to write directly to the file on the disk and to bypass the cache.

**FAT Cache Threshold:** FAT provides a way to specify a cache thresh-



hold parameter, which allows you to tell the FAT file system how large an I/O operation to cache. It is defined as a number of 512 KB sectors between 1 and 128. This number should be based on the size of the cache and the size of the I/O blocks being transferred. It should be as large as the largest block of data that is processed, and should not exceed one-fourth of the total cache size. For example, if the cache size is 64 KB, set the cache threshold at 32; for 128 KB, use 64; and for more than 128 KB, use 128. These numbers are guidelines for improving the performance of application I/O and application loads. The actual setting will vary, depending on the actual application mix. For example, if OS/2 is used only for database operations with I/O in 8 KB blocks, then the optimal threshold setting is probably 16. This would cover the maximum I/O size and enable the greatest number of data blocks to be cached.

The size of the cache threshold is specified by a parameter in the DISKCACHE statement in the CONFIG.SYS file. If no value is specified, a default of 4 is used. For example, here is the DISKCACHE statement for a 256 KB cache and a cache threshold of 32 KB (the null field

between the two commas pertains to lazy writing).

```
DISKCACHE=256, ,32
```

## Lazy Writing

The lazy write capability is available in OS/2 2.0 for both the FAT and HPFS file systems. *Lazy writing* is a way to delay physical disk I/O writes until a time when the write operation will have little impact on the rest of OS/2. The application writes records into the cache and regains control immediately. When the disk is not busy with other tasks, or when the cache has too many updated (also called *dirty*) blocks in it, the physical I/O will be done asynchronously by another processing thread. Lazy writing can improve the performance of write operations by 20% to 40%, depending on the type of I/O and the hardware being used.

**HPFS:** To start the lazy writer and allocate its associated control blocks in previous releases of OS/2, there had to be a RUN=CACHE.EXE command in CONFIG.SYS. This created extra system overhead. In OS/2 2.0, during system startup, the STARTLW.DLL statement in the OS2.INI file loads a DLL that starts the lazy writing capability under HPFS. The default is to

start the lazy writer. The lazy writer can be turned on or off any time by issuing the `CACHE` command from a command prompt or a command file. You also can add the statement `RUN=CACHE.EXE` to `CONFIG.SYS` to turn lazy writing off when the system is booted.

**FAT:** To use the lazy writer under FAT, put the `LW` parameter on the `DISKCACHE` statement in the `CONFIG.SYS` file. The following statement turns on the lazy writer:

```
DISKCACHE=256, LW
```

Unlike HPFS, once lazy writing is turned on under FAT, there is no way to turn it off unless you change the `CONFIG.SYS` file and reboot the computer.

There are three parameters for tuning the lazy writing process, as follows:

**MAXAGE:** The `MAXAGE` parameter specifies the maximum amount of time in milliseconds that an updated data block can remain in the cache before it is written to disk. The default is five seconds. For critical data, this parameter can be reduced to a half second or less, but be aware that the overhead of the lazy writer can degrade performance if the `MAXAGE` parameter is too low. It would be better if the application can specify file write-through when the critical data file is opened. This is done by setting a parameter bit in the `DosOpen` command in the application. This bit tells the HPFS or FAT to read from and write to the physical disk directly, instead of going through the cache. Although this operation is slower than using the cache, it provides the most data security.

**DISKIDLE:** The `DISKIDLE` parameter specifies the amount of time in milliseconds that the disk is idle. If the disk is idle for this amount of

time, updated data blocks in the cache are written to the disk. The default time is one second. Anything less than that may cause interference with the rest of the system. In special cases, a smaller number may be better for performance reasons. If much read I/O is being performed and operator response time is critical, you may want to increase the disk idle time.

**BUFFERIDLE:** If no records are passed into or out of the file system buffer during the time (in milliseconds) specified by the value in `BUFFERIDLE`, then OS/2 starts to write updated records from the cache to the disk. The default is a half second.

All these parameters have defaults for optimum performance in general environments. They should be tuned only for specific circumstances where the operating procedures are known and the level of data security can be defined. The parameter settings should not be reduced for systems that have much ongoing swap activity. Reducing the values would cause contention problems on the disk between the cache program and the page swapper, and would degrade the computer's overall performance.

The `MAXAGE`, `DISKIDLE`, and `BUFFERIDLE` parameters are specified in the `CACHE` command for the HPFS file system; they can be changed while the system is running. There is no way to change these parameters for the FAT file system.

### OS/2 System Internals

This section discusses three aspects of OS/2 2.0: flat memory model, paging systems, and system resources.

#### Flat Memory Model

Previous versions of OS/2 and all versions of DOS require application programmers to treat their data as artificially sized units called *segments*.

This restriction forced developers to adopt techniques for managing these segments, and resulted in programs that were complex and not compatible with other platforms.

OS/2 2.0 allows application developers to allocate data objects of up to 448 MB. Additionally, it gives developers extensive control over how the system manages the real hardware resources devoted to these objects. Developers are free to use the large data addressing capability of the system – and without excessive system resources – to manage these objects.

OS/2 2.0 also allows application programs to be much larger than in previous versions. The artificial limits imposed by OS/2 1.X and by DOS, in some cases, forced developers to segment their application code in ways that were not compatible with the application logic.

The 80386 and 80486 processors implement memory management using a flat object addressing scheme. Objects can range from 1 byte to 4 GB, and there can be up to 16,384 objects. OS/2 2.0 uses a single object of up to 4 GB. In this space, application programs can address up to 512 MB, but 64 MB are reserved for shared objects. Applications can use the remaining 448 MB for private code and data. (This number may be reduced to 384 KB because of protected DLL support.) Each application receives its own copy of this address space. Because it uses a single large memory object, this implementation is called the *flat memory model*. Using this model, application programs no longer need to load or change any of the segment registers, which simplifies application programming and significantly enhances performance.

#### Paging System

In OS/2 1.3, data is managed in units called *segments*. The developer deter-

mines the sizes of these segments up to 64 KB. Managing segments of various sizes, both in memory and on disk, requires that the operating system do the following:

- Compress real storage when a request for a segment was generated that exceeded the contiguous storage available in the system.
- Use many different sizes of space allocation to manage disk space for the segments swapped out. This frequently resulted in excessive space allocations for swapped-out memory.

The need for these functions is eliminated in OS/2 2.0, which manages storage and swap space in fixed-size units called *pages*. If there is a need for a page of memory or disk space, OS/2 can either find a free page immediately or make one available easily. The application requires no programming support. Following is the process required to fulfill a request for a new page of memory:

1. Determine whether a free page of memory exists.
2. If none exists, find the Least Recently Used (LRU) page.
3. If the swap file is too small to hold the page, increase its size.
4. Write the LRU page to disk. If more than one page exists in the LRU list, a maximum of eight pages will be written to the disk.
5. Fulfill the request using either the free page found or the page written to disk.

If an application tries to access a page of memory that has been swapped to disk, OS/2 allocates a free page as above, then reads the swapped page from disk into the free page.

If an application frees a page, that page is added to the list of free pages (if the page is in memory), or the disk space occupied by the page is

freed (if the page has been swapped to disk). Periodically, the swap file is examined to see whether it can be made smaller, which will be possible when all swapped pages placed at the end of the file have been freed. This is a significant change from OS/2 1.X, where the swap file would never shrink.

### Memory Allocation

The term *memory object* identifies memory allocated in OS/2 2.0. Memory objects can be any size between 1 byte and 448 MB; once allocated, their size cannot be changed. They are allocated in units of one page, which equals 4096 bytes, and each page is aligned on a 4096-byte boundary.

Memory objects can be allocated as shared objects, in which case they are allocated from the address space between 448 MB and 512 MB, which is the space reserved for shared objects.

Within a memory object, each individual page can have the following characteristics:

- Real memory committed to it (Without this attribute, an attempt to read or write to the specified page results in a page fault exception.)
- Read only (An attempt to store into a page that has the read-only attribute results in a page fault exception.)
- Read/write
- Contain code that is executed by the processor (This attribute implies read-only.)
- Guard page (This is a page of memory that has not yet been allocated, but has been reserved in case more data must be saved than can fit in the page. An attempt to store into a page with this attribute results in a guard-page fault exception. The storage will succeed if

real memory was committed to the page before the storage attempt.)

When coding an application, developers should allocate memory using the `DosAllocMem` API for all required memory. Memory can be allocated as either committed or uncommitted. *Committed memory* means that at the time the memory is allocated, either actual RAM or space in the swap file is made available. *Uncommitted memory* means that the page table entries and other control block information needed to define and access memory are created, but neither RAM nor swap file space is allocated. When the memory is needed, the application issues an API to commit the defined memory. Space is then made available in either RAM or the swap file. In OS/2, it is far better for applications to allocate memory as uncommitted, then commit it as needed. This reduces the time necessary to allocate the memory, and can reduce the possibility of having to swap data between RAM and the swap file.

It is important to note that the PM memory and heap management APIs in OS/2 1.X are supported only for 16-bit applications in OS/2 2.0. If a 16-bit application is being updated, the developer should replace the PM memory and the heap APIs with the `DosAllocMem` and `DosSubAllocMem` APIs to get a true 32-bit application.

### System Resources

Figures 5 and 6 list the major OS/2 and Presentation Manager resources and their limits. The figures compare OS/2 1.3 and OS/2 2.0, and highlight the much larger capacities built into OS/2 2.0.

Hardware resources and limits in OS/2 2.0 have not changed significantly from those in OS/2 1.3. The major differences are as follows:

1. Asynchronous ports are increased to four for both Extended Industry

Resource	OS/2 1.3	OS/2 2.0
Full-Screen Sessions (including Multiple Virtual DOS Machines or MVDMs)	16 limit	255 limit
	12 available	255 available
Virtual Input/Output (VIO) Sessions (includes MVDMs)	16 limit	255 limit
	12 available	252 available
MVDM Sessions	1 limit	255 limit
	1 available	252 available
Processes	511 limit	4,096 limit
	504 available	4,065 available
Threads per OS/2 System	512 limit	4,095 limit
	483 available	4,065 available
Threads per Process	54 limit	4,095 limit
	54 available	4,065 available
File Handles per OS/2 System	65,536 limit	65,536 limit
	65,476 available	65,470 available
File Handles per Process	32,766 limit	32,766 limit
	32,761 available	32,761 available
Global Semaphores (per OS/2 System)	512	64,000
Private Semaphores (per Process)	128	64,000
Addressable Memory per System	16 MB	4 GB
Addressable Memory per Process	16 MB	512 MB

*Note:* These numbers assume that there is sufficient physical memory and disk memory. They also assume that both HPFS and FAT are installed and that the lazy writer is active.

These numbers represent available resources in OS/2. If these limits are reached, performance will degrade.

**Figure 5. OS/2 Resources and Limits**

Resource	OS/2 1.3	OS/2 2.0
Presentation Manager Sessions	16 limit	255 limit
	14 available	252 available
String Atoms	16 KB limit	16 KB limit
Integer Atoms	48 KB limit	48 KB limit
Fonts	254 limit	254 limit
Heap Size	64 KB limit	64 KB limit
Logical Color Table Size	31 KB limit	31 KB limit
Window Objects	1,400 limit	1,400 limit

**Figure 6. Presentation Manager Resources and Limits**

Standard Architecture (EISA)- and Micro Channel-based computers.

2. Three internal diskette drives are now supported.
3. CD-ROM device driver support has been implemented.
4. Int13 disk drive support via CBIOS has been implemented. This allows users to use hard disks for which there is currently no device driver. It is meant as a temporary support mechanism until an OS/2 device driver is available.

*Ron Morrill joined IBM in New York as a systems engineer on the City College team in 1968. Since that time he has supported TSO, APL, CP67, and other time-sharing systems. He has been a system support representative for HONE and a systems engineering manager supporting Texas Instruments in Dallas, Texas. He is currently a senior marketing support representative supporting OS/2.*

*Ron Cadima is an advisory programmer in the OS/2 Performance Support group in the IBM Personal Systems Programming Center in Boca Raton, Florida. In his 24 years with IBM, Ron has worked on operating systems development, production and process control applications, communications with point-of-sale systems, and OS/2 system performance. He was the lead performance analyst for OS/2 1.3, and he currently works with customers, independent software vendors, and others in designing OS/2 applications and tuning OS/2 systems for performance.*

# OS/2 2.0 Application Support

**Ron Morrill**  
**IBM Corporation**  
**Roanoke, Texas**  
 and  
**Ron Cadima**  
**IBM Corporation**  
**Boca Raton, Florida**

*Application support in OS/2 2.0 has been greatly improved over OS/2 1.X. Under OS/2 2.0, all applications have the advantages of improved file systems such as the File Allocation Table (FAT) and High Performance File System (HPFS), better implementation of system services, better use of hardware resources, and the replacement of segment swapping with paging.*

*This article covers 16-bit OS/2 applications, 32-bit OS/2 applications, and DOS/Windows applications.*

## 16-Bit OS/2 Applications

**S**ixteen-bit applications are written for OS/2 1.X. They are written to conform to the segmented memory architecture of the 80286 processor. Each subsequent OS/2 1.X version contains the system calls defined in previous OS/2 1.X versions. In OS/2 2.0, all 16-bit applications run as they did under OS/2 1.X. However, changes have been made in the way OS/2 2.0 performs certain services. This article describes these changes for application developers.

### Memory Allocation

In OS/2 1.X, when a small segment of memory was allocated to an object, the segment was given only the amount of memory it needed – the real memory assigned to the object plus a small amount of overhead. However, in OS/2 2.0 the minimum unit of memory allocated to any object, no matter how small, is one page, which is 4,096 bytes. Therefore, if a 16-bit application requests a small segment of memory, one page is assigned. This minimum size

affects 16-bit applications that allocate many small segments.

If an application requests a large number of segments, as applications often do, large amounts of real memory are wasted. Some examples follow:

- Assume an application requests 10 segments of 100 bytes each. Under OS/2 1.X the requests are allocated a total of 1,000 bytes of real memory, whereas under OS/2 2.0 the requests are allocated  $10 \times 4,096 = 40,960$  bytes of real memory.
- If an application requests 20 segments of 5,000 bytes each, under OS/2 1.X the requests are allocated 100,000 bytes of real memory; under OS/2 2.0, the requests are allocated 163,840 bytes of real memory. Each 5,000-byte segment is allocated two 4,096-byte pages, or 8,192 bytes of memory; 20 segments  $\times$  8,192 bytes per segment = 163,840 bytes.

The System Performance Monitor/2 (SPM/2) tool can identify applica-

tions that request large numbers of segments. One way to decrease the memory usage of such an application is to rewrite its memory management routines so that they fit better with the OS/2 2.0 allocation method. Applications can do this by using `DosSubAlloc` or the C library routine `malloc` to allocate the small data areas.

Another way to reduce memory usage is to use the 16-bit pack facility in OS/2 2.0. This option packs 16-bit private and shared code and read-only data segments. The default is to do packing automatically when the application is loaded. No action is required by the user or the application.

Users can change the default by specifying a `NOPACK` option for the `MEMMAN` statement in the `CONFIG.SYS` file. For example, the default statement would change from `MEMMAN=SWAP, PROTECT` to `MEMMAN=SWAP, PROTECT, NOPACK`. This may yield a small increase in performance, but it also can increase the size of the swap file. Measure any gain in performance and compare that to the possible growth in the size of the swap file.

The change in memory segment allocation also causes a potential incompatibility in analyzing problems. Because OS/2 1.X allocated the exact number of bytes requested by a segment allocation request, an attempt to store data into, or fetch data from, memory that is past the end of the segment resulted in a trap 000D. Under OS/2 2.0, segment allocations are performed in multiples of 4,096 bytes. If you allocate a 2,000-byte segment, it is placed in a 4,096-byte page. If the application now tries to access byte 2,001, the hardware will not generate a trap 000D exception as it did for OS/2 1.X, because byte 2,001 is within the 4,096-byte page.

In OS/2 2.0, you will get the exception only if the size of the segment was a multiple of 4,096 bytes and if the application tried to access something outside that size. For example, if the application allocates a 4,096-byte segment, then tries to access byte 4,097, the hardware exception will be generated.

### 32-Bit OS/2 Applications

New Application Programming Interfaces (APIs) support 32-bit applications. While many APIs are the same as their 16-bit counterparts, significant changes have been made in many others.

#### Memory Management

OS/2 2.0 uses the flat memory model, and 32-bit applications are written to conform to this model. It allows 32-bit applications to allocate memory objects rather than memory segments as in OS/2 1.X versions. These objects can be any size from 1 byte to 448 MB. Applications can realize significant performance gains simply by using large memory objects. In addition, by controlling their memory objects, applications allow OS/2 to make better use of real machine resources.

Allocation of memory objects can be a two-step process. In step one, the application reserves address space for the memory object. This creates the entries in the page tables that define the memory object, but it will not allocate any RAM or space in the swap file for the memory object. The reserved address space is called *uncommitted memory*. Step two of the process occurs when the application needs to use that memory. In this step, the application issues an API, `DosSetMem`, to commit that memory. Then OS/2 allocates actual RAM or swap file space for the memory object. When a page is first committed, it is filled with binary zeros.

Here are two examples of using this process:

1. Suppose an application uses a memory object as a linear array of elements. The application stores data into the first element, then the second, and so on. As it is saving data, the application checks whether the next array element can fit into the real memory already committed. If the next array element cannot fit, the application issues a command to commit the next section of allocated memory.
2. Suppose an application uses a memory object as a random array of elements. When the application wants to store data into an array element, it determines the specific element by using an algorithm such as a key value or a hashing technique. It then checks whether the corresponding page of the memory object has already been committed. If not, the application commits the page.

In both cases, the application may later determine that a page containing certain elements of the array is no longer needed. It can then decommit the corresponding page of the memory object. Of course, if the entire memory object is no longer needed, it can be freed using `DosFreeMem`.

Significant advantage is gained in the above scenario only if the memory objects are of substantial size or are large in number. Remember, to manage objects of less than page size, use the C compiler's `malloc` or `DosSubAlloc`.

It is far more efficient to allocate a large memory object with uncommitted memory, then suballocate it and commit the individual memory objects as needed. This ensures that you are using only the amount of memory that is needed. At the same time, it can reduce the size of the swap file if more application memory

is being used than is physically available in the computer.

Optimum disk Input/Output (I/O) is achieved when it is performed in blocks of 32 KB and 64 KB. When allocating either I/O memory objects or objects that will be paged, it is best to allocate them to take advantage of the optimum I/O block size.

The heap management APIs that existed in Presentation Manager (PM) in OS/2 1.X exist in OS/2 2.0 only for 16-bit applications. It is recommended that the `DosAllocMem` and `DosSubAlloc` APIs be used. This should be a simple change because of the flat memory model, and will yield performance gains of 5% to 15% over the OS/2 1.X PM heap manager.

#### Stack Management

A 32-bit application uses memory objects for its stack. A stack can be up to 448 MB. Real memory is not assigned to a stack until the application requires it. This is done by committing only the last two pages of a stack (stacks are used from the top down). The lower of the two pages is marked as a *guard page*, a page of uncommitted memory that is not allocated until the data in the previous page overflows. When the application starts to use this page, an exception is generated. As a result of this exception, the next lower page is committed and marked as a guard page. Thus, memory is gradually committed as the application requires it. This process ends when the memory object used for the stack is completely committed.

Because stacks in OS/2 2.0 are essentially unlimited in size, application programmers can make more use of automatic variables. Automatic variables are allocated when a function starts and are automatically freed when that function ends. Allocating

and freeing automatic variables is extremely efficient, because the memory already exists in the stack, and no new base address must be loaded to access the data.

### Semaphores

Semaphores in 32-bit applications have significantly changed. There are now two classes and three types of semaphores. The two classes are as follows:

- **Private** semaphores are used within a single process. An application can have up to 64,000 private semaphores.
- **Shared** semaphores can be shared by all processes. A total of 64,000 shared semaphores can be created by all processes in the system.

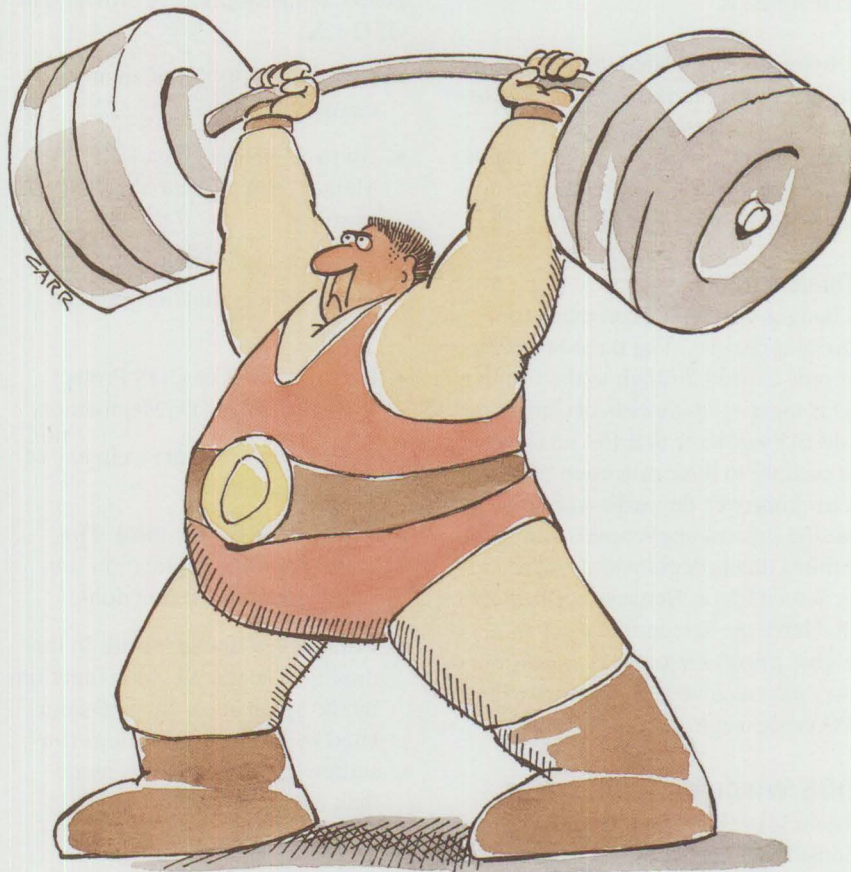
The three types of semaphores are as follows:

- **Event** semaphores allow one task to notify another task that something has happened. Examples of events include when file I/O is complete; database retrieval is complete; the user has requested that a function be executed.
- **Mutex** semaphores are designed to protect a common resource, such as a common data area used by more than one thread. A mutex semaphore protects that area from simultaneous update by more than one thread.
- **MuxWait** semaphores allow a thread to wait for one or more of the above types of semaphores to be posted.

All semaphore processing has been rewritten for the 32-bit application interface. Semaphore processing is much faster in the new application interface.

### Thread Management

The ability to start and control threads has been enhanced in OS/2 2.0. This makes threads easier to use



and results in improved overall system performance.

OS/2 2.0 supports more threads – up to 4,096 threads for the entire system (any subset of which can be allocated to a process). Formerly, a process could start no more than 53 threads. The amount of real memory required to support each thread is reduced.

Stack allocation and freeing is done automatically by `DosCreateThread`. This eliminates the requirement that OS/2 1.X places on the creating code to allocate the new thread's stack. (In OS/2 1.X, there was no supported way to free the stack when the thread ended.)

Threads can be created in a suspended state. This enables an applica-

tion to create threads that are ready to perform their required functions before they are needed. Applications may therefore be more responsive to requests when the required function needs to be executed.

At any point, the execution of a thread can be suspended. This gives an application ultimate control over the allocation of processor resources to its threads.

Obviously, suspended threads can be resumed and threads created in a suspended state can be started. At any time, applications can find out about the current state of a thread by using the `DosGetInfoBlocks` API.

### Thread Priority

Changes also have been made to thread priorities. The thread priority is now carried through to the file I/O. This means that threads performing file I/O will have that I/O serviced according to their execution priority. This improves the performance of timing-critical applications and applications that process with foreground or screen focus. Because applications that have the screen focus get a higher priority in process execution, they also receive higher priority for I/O processing.

### DOS/Windows Applications

The rest of this article describes considerations for running DOS applications under OS/2 2.0. These considerations also apply when running Windows applications, since Windows applications run under the control of a DOS session. Not all aspects of this support are discussed here. For additional information, refer to *OS/2 2.0, Volume 2: DOS and Windows Environment* (GG24-3731).

Users can run up to 240 concurrent DOS application sessions. Like all other OS/2 2.0 sessions, every DOS session has preemptive multitasking,

a protected environment for each application, and the performance advantages of the new FAT and HPFS file systems. Generally, if an individual DOS application "hangs" its DOS session, that session terminates without affecting other DOS or OS/2 sessions.

Each DOS session can be provided with much more memory than the DOS Compatibility Box provided in OS/2 1.X:

- More than 630 KB of standard memory
- Up to 32 MB of Expanded Memory Specification (EMS) memory
- Up to 16 MB of Extended Memory Specification (XMS) memory
- Up to 512 MB of DOS Protect Mode Interface (DPMI) memory

Each DOS session can run in any of the following modes:

- **Full-screen foreground:** This mode gives the application full access to all video functions.
- **Full-screen background:** In this mode, all access to video functions by the application must be simulated by OS/2. Applications run somewhat slower in this mode than in full-screen foreground mode.
- **In a DOS window on the Presentation Manager (PM) desktop:** Cut and paste are supported in this mode. Users can cut any part of the window to the clipboard, and conversely can paste text information from the clipboard to the DOS application. All access to video functions by the application must be simulated by OS/2. The application runs somewhat slower in this mode than in full-screen foreground mode. In this mode, an application is prevented from executing if it performs XGA or 8514/A graphics

writes to the video display. There may be a slowdown in your computer's performance if you execute multiple full-video applications concurrently.

### DOS Memory Extenders

DOS sessions assume they have total control over the hardware, so they consume as much resource as they are given. If you allow each of several DOS sessions to consume large amounts of EMS, XMS, or DPMI memory, you can run out of memory or disk space.

OS/2 2.0 lets you specify the maximum values for EMS and XMS memory to be used by *all* DOS sessions and the default values for *each* session. OS/2 2.0 also lets you specify the maximum amount of DPMI memory for *each* session. The values for all sessions are specified in the `CONFIG.SYS` file.

To enable DOS sessions to use EMS memory, load the Virtual Extended Memory Manager (VEMM) device driver by specifying the following `DEVICE=` statement in `CONFIG.SYS` (which assumes OS/2 2.0 is installed on the C: drive):

```
DEVICE=C:\OS2\MDOS\VEMM.SYS
        4096, 2048
```

Here, 4096 represents the total amount of EMS memory available to all DOS sessions, and 2048 represents the default amount of EMS memory available to each DOS session.

To enable DOS sessions to use XMS memory, load the Virtual Extended Memory Support (VXMS) device driver by specifying the following statement in `CONFIG.SYS`:

```
DEVICE=C:\OS2\MDOS\VXMS.SYS
        4096, 2048
```

As with the EMS driver, 4096 represents the total amount of XMS mem-



ory available to all DOS sessions and 2048 represents the default amount of XMS memory available to each DOS session. The VXMS device driver statement should follow the VEMM statement in CONFIG.SYS to prevent conflicts between the two drivers.

To enable DOS sessions to use DPPI memory, load the Virtual DOS Protection Extension (VDPX) and Virtual DOS Protect Mode Interface (VDPMI) device drivers by specifying the following statements in CONFIG.SYS:

```
DEVICE=C:\OS2\MDOS\VDPX.SYS
DEVICE=C:\OS2\MDOS\VDPMI.SYS
```

No parameters are associated with these drivers. Because there is no system-wide limitation on DPPI memory, take care when specifying the DPPI maximum for each DOS session. These same settings apply to Windows sessions. However, there are some additional considerations for Windows sessions depending upon the mode in which Windows applications are run.

All Windows applications can be executed in a common Windows session, either in a full-screen mode using the Windows Program Manager, or in windowed or full-screen sessions without the Program Manager. The latter are "seamless" sessions where the Windows applications are started from an icon on the desktop or in an OS/2 folder. When a Windows application is defined, the user can choose (in the Settings portion of the session binder) whether to run the individual Windows application in full screen or in a window. If the application is to be run in a window, the user can choose between a separate session or a common session.

Separate sessions provide more security and can improve system responsiveness. For example, when you run a Microsoft Word for Windows

macro in a native Windows environment, you may notice that it is difficult to change the screen focus and to switch to another application. This same phenomenon occurs when you run this application in a common WIN-OS2 session. However, if you run Word for Windows in a separate WIN-OS2 session, you can switch between applications without delay. The separate session also improves data security. If the application running in that session has an unrecoverable error, that session fails, but it does not affect any other active session.

The drawbacks of having separate WIN-OS2 sessions are that they require a lot of memory and are slow to start up. Starting a separate WIN-OS2 session is equivalent to booting a computer with DOS, starting Windows, then loading the Windows application. In addition, memory is needed for each of these "computers." The trade-off of having separate WIN-OS2 sessions, therefore, is to gain session security, performance, and responsiveness, while seeing an increase in the use of computer resources and slower startup of applications.

With common WIN-OS2 sessions, session security is lost and applications run as they would in a Windows environment. However, startup performance and system resource utilization improve. The first common session Windows application that you load is equivalent to booting DOS, starting Windows, then loading the Windows application. But when subsequent Windows applications are loaded, you need only load the application.

When the required amounts of XMS, EMS, and DPPI memory are defined for a common session, the total must equal the memory requirements of all the Windows applications that will run concurrently in the common

WIN-OS2 session. For example, if you run three Windows applications in a common WIN-OS2 session, and each application requires 2 MB of DPPI memory, you must allocate 6 MB of DPPI memory for the common WIN-OS2 session.

As soon as they are loaded, certain Windows applications will try to access all the DPPI memory. Depending on where these applications are loaded, this can cause a sudden large growth in the swap file and can adversely affect performance. It is probably best to start these applications after all others have been started.

If there is a Windows application that you always use in the common WIN-OS2 session, it is a good idea to start that application right after OS/2 2.0 is started. To start an application immediately, simply place the application into the Workplace Shell's Startup folder. To load another Windows application after this application has been loaded, the DOS and Windows environment will not load again, and the application loads much faster.

Another way to enhance the performance of Windows applications is to use a "private" clipboard and Dynamic Data Exchange (DDE) support. (The default is to use public support.) This means that DOS, Windows, and OS/2 applications can communicate through DDE and the clipboard, if they use the same protocols. If only the Windows applications in a common Windows session need DDE or clipboard support, they should use the private support. To specify private support, perform the following steps after starting the common Windows session:

1. Restore the clipboard program in the WIN-OS2 session. Select Options from the action bar and deselect the Public Clipboard option.

2. Select the DDE Interchange Agent from the WIN-OS2 session and close it.
3. Optionally, bring up the OS/2 Window List and close the Data Update and Clipboard processes displayed in it.

A common WIN-OS2 session can run with one or more separate WIN-OS2 sessions. This allows greater flexibility in balancing memory resource usage, OS/2 resource usage, and OS/2 system performance.

### DOS Properties

The DOS Properties feature of Multiple Virtual DOS Machines (MVDM) gives the user the ability to selectively configure and tune the Virtual DOS Session environment to meet the requirements of particular applications. Because some DOS applications require certain features while other applications operate better without them, a Virtual DOS Session can be individually configured to provide the optimum execution environment for the application.

The user can set DOS Properties when adding an application to a group on the desktop, or in certain cases, during execution while an application is running within the Virtual DOS Session. In the case where a Virtual DOS Session is created by another process using a `DosStartSession()` function call, a buffer can be provided that contains the required DOS Properties and their values.

The following discussion covers only those DOS Properties that directly affect OS/2 or Virtual DOS Session performance. They let users exercise extensive control over the performance characteristics of the Virtual DOS Session.

DOS Properties can be changed in either of two ways:

- Settings that are adjustable only when a Virtual DOS Session is created can be changed only before starting the Virtual DOS Session. If the DOS application is placed into a folder in the Workplace Shell, the DOS Properties can be changed by selecting the Properties option in the application's context menu.
- Settings that are adjustable at any time can be set in the manner described above, or they can be changed while an application is running in a Virtual DOS Session.

Certain settings can be changed both ways. For some DOS applications that exist at the time OS/2 2.0 is installed, the installation process automatically creates the DOS Properties settings. Updates and new DOS Properties will be made available on bulletin board systems, such as CompuServe or IBM bulletin boards.

The `DATABASE.TXT` file in the `\OS2\INSTALL` subdirectory contains a list of all the applications tested by IBM with OS/2 2.0 and their settings. You can modify this file to change DOS settings or to add applications. Once the file is updated, it must be run through an application called `PARSEDB`, also in `\OS2\INSTALL`. This program generates a `.DAT` file; you must give this file a name other than the name of the existing `.DAT` file, `DATABASE.DAT`. The OS/2 migration utility uses this new `.DAT` file to migrate and install DOS, Windows, and OS/2 applications.

Even the most innocent of the DOS settings can affect your OS/2 system's performance and resource utilization. For example, the `DOS_LASTDRIVE` setting defines the letter of the last drive that can be accessed; this setting affects the computer because every drive letter costs approximately 100 bytes of memory. This may seem insignificant, but it can mean

the difference between paging and not paging memory.

The rest of this article covers the most important DOS and Windows settings. For a description of all the DOS and Windows settings, refer to *OS/2 2.0, Volume 2: DOS and Windows Environment*.

Figure 1 shows settings to control how screen I/O operations function within a Virtual DOS Session. The settings in Figure 2 affect the virtual hardware environment provided by the Virtual DOS Session. Figure 3 shows settings affecting the behavior of the DOS emulation environment within a Virtual DOS Session. Figure 4 shows settings that affect the behavior of the EMS, XMS, and DPMI memory extenders, if used in the Virtual DOS Session. Figure 5 shows settings that affect file I/O operations within Virtual DOS Sessions. Figure 6 shows settings that affect the screen I/O behavior of Virtual DOS Sessions when running in windowed mode on the Presentation Manager desktop.

### Other Considerations

Tools are under development for tuning and analyzing OS/2 applications and performance. The first tool that will be available is `SPM/2`. This tool can be used to report processor, memory, and disk resource usage, application and OS/2 working set sizes, disk and file system utilizations, and processor utilization.

Remember, there are 32-bit versions of existing OS/2 APIs as well as new 32-bit APIs that support new functions like the 32-bit flat memory model. Whenever an application is developed, these APIs should be used instead of the old 16-bit ones. Using these APIs can produce significant performance gains over a segmented application with a minimum of a 20% gain. If the design takes

advantage of OS/2's features, gains up to 60% are possible.

The actual gain depends on the functions that the application performs and the resources required. The more an application is memory-intensive, with large memory object processing, the greater the potential for performance enhancements.

To attain the higher gain, design applications to take advantage of the flat memory model and multi-threaded execution. At a minimum, each application should contain three separate threads: one to perform I/O, one for mainline processing, and one

for initialization and error handling. Developers should code to conform to the PM model and take advantage of the common file and font dialogs that are supplied with OS/2. This not only saves space and coding complexity in the application, but it ensures that the user will work with the same interface for all applications including the OS/2 system.

Even though a paging system is more efficient than a segmented system, and even though the flat memory model enables greater use of system resources, developers must still be aware of the system resources that they are using and minimize them as

much as possible. This means reducing the working set of the application environment to minimize the amount of paging that takes place and to maximize the performance of the system. With this in mind, limit the number of DLLs that the application contains to major function areas. One DLL may be for initialization, termination, and error processing. Others should be built based on related functions that are used in conjunction with each other.

When setting up the system, OS/2 uses only the numbers of fonts, code pages, and device drivers that are required by the environment. Be sure

#### 8514/A & XGA I/O Trapping

**Function:** When this setting is set to on, it provides faster, unrestricted access to 8514/A display adapter hardware.

**Advantages:** It achieves higher performance for 8514/A applications and eliminates the overhead of the 1 MB 8514/A virtual video buffer normally allocated for each Virtual DOS Session.

**Drawbacks:** Screen-switching away from the application results in immediate freezing of the application, and the system may not be able to reliably switch back because the screen image may not be correct. This can be overcome by setting Video Screen-Switch Notification that notifies applications to redraw their own screen images. Not all applications can take advantage of the notification. An application with this setting enabled cannot be run in a windowed mode or copied to the clipboard, because there is incomplete information about the state of the screen buffer.

**Default:** Off.

**Settable:** At Virtual DOS Session creation only.

**Example:** When executing Windows 3.0 with the 8514/A display driver, certain operations, such as painting dithered backgrounds, will run significantly faster.

#### Video Mode Restriction

**Function:** This setting extends the 640 KB DOS address space by limiting video-mode support.

**Advantages:** For text-based or Color Graphics Array (CGA)-based applications, the video memory normally reserved for high-resolution graphics modes just above 640 KB can be remapped to conventional memory. This frees an additional 64 KB (or 96 KB, depending on graphics mode) for applications. This is valuable for applications that do not take advantage of EMS or XMS memory extenders.

**Drawbacks:** It is not possible to completely hide the fact that the video adapter is capable of high-resolution graphics; some applications may attempt to enable those modes and to use the memory above 640 KB as video memory, inadvertently corrupting the application that is using that area. Exercise care when using this feature.

**Default:** None. There are three possible settings:

- None, which defaults to the OS/2 definition.
- CGA modes only (adds 96 KB). Only applications that support this mode can run in it.
- Monochrome modes only (adds 64 KB).

**Settable:** At Virtual DOS Session creation only.

Figure 1. Video Settings (continued)

**Video: On-Demand Memory Allocation**

**Function:** This setting reduces swap space requirements for full-screen Virtual DOS Sessions.

**Advantages:** This setting allows a full-screen Virtual DOS Session to run without preallocating a virtual video buffer for high-resolution graphics modes. This setting does not prevent execution of graphics applications; allocation of the buffer is simply delayed until it is needed. This can save a substantial amount of memory and swap file space, which could be important when memory is constrained.

**Drawbacks:** If the allocation of a virtual video buffer for a full-screen Virtual DOS Session fails at the time the application changes video modes, the system must freeze the session and switch control back to the shell. Unless the user can free memory from another session, the user may be unable to get the DOS application running again. This is a concern if the application contains unsaved data.

**Default:** Off.

**Settable:** At any time. Becomes effective the next time the session switches to a full screen.

**Video: Retrace Emulation**

**Function:** This setting simulates the video retrace status port to provide faster access.

**Advantages:** DOS applications that poll the video retrace status port often write to the screen only during the retrace interval, even though it is safe (on EGA and VGA adapters) to draw at any time without causing "snow" interference. This feature causes most applications to write to the screen more often, and compensates for the performance drag imposed by monitoring the port in the first place.

**Drawbacks:** Some applications may poll the port in such a way that overall performance is worse; this is sometimes true of applications that draw only during vertical (not horizontal) retrace. Unfortunately, while turning off trace emulation will restore performance, there is a risk that screen-switching will not be as reliable.

**Default:** On. Reliable screen-switching has higher priority over the minority of applications that experience some drag in performance.

**Settable:** At any time. This enables users to experiment with different settings if a performance problem occurs.

**Video: ROM Emulation**

**Function:** This setting emulates selected INT 10h Read-Only Memory (ROM) video functions.

**Advantages:** This setting provides faster output for selected video functions than ROM services typically provide. It has a dramatic effect on the performance of functions that are in a window.

**Drawbacks:** Some ROM may offer enhanced services that are not included in the emulation. Applications that rely on these services may not execute correctly.

**Default:** On. Because the INT 10h ROM video services are well documented, incompatibilities are unlikely and the performance benefits of using the emulation are quite significant.

**Settable:** At any time. This allows the user to experiment in the event of a compatibility problem.

**Video: Screen-Switch Notification**

**Function:** This setting notifies a DOS application of a switch to or from full-screen mode.

**Advantages:** This setting allows applications that monitor this notification to redraw their screens as needed. This may be necessary for some video adapters with modes (and applications that use those modes) that are not fully supported by the OS/2 video driver or those that are slightly incompatible. It is also valuable in situations where an OS/2 video driver has not allocated a virtual video buffer (see "8514/A & XGA I/O Trapping" above).

**Drawbacks:** When used indiscriminately, this feature can cause unnecessary and time-consuming screen redrawing. For standard mono/CGA/EGA/VGA video modes, the OS/2 video driver should be able to restore application screens without assistance.

**Default:** Off. For standard hardware and standard video modes, this feature is not necessary.

**Settable:** At any time. This allows the user to experiment when a compatibility problem occurs.

**Example:** Windows 2.X and 3.X understand this notification and redraw themselves accordingly.

Figure 1. Video Settings

**Break**

**Function:** Determines whether the operating system checks for the keystrokes Ctrl+Break or Ctrl+C while a program is processing. If the Break setting is enabled, pressing these keystrokes causes OS/2 to terminate the current application.

**Default:** Off. The operating system checks only for Break key combinations during standard input or output operations. Programs run more slowly when BREAK=ON.

**Settable:** At initialization time or at Virtual DOS Session creation. It is not possible to change the Break setting after a program is running in a Virtual DOS Session.

**Example:** OS/2 checks the Break key combinations when there is input from the keyboard to the program or output from the program to the screen or a printer. It does not, however, check while the program is processing information (for example, while a program is being compiled).

**Copy ROM to RAM**

**Function:** Enabling this setting causes OS/2 to copy ROM and run the copy in 32-bit RAM. With this setting enabled, BIOS operations run faster and system utilities can patch BIOS – that is, change instructions and code in BIOS.

**Default:** Off.

**Settable:** At Virtual DOS Session creation only.

**Examples:** This setting is useful if debugging the kernel. Copying ROM to RAM allows normal breakpoints to be set in ROM and allows stepping over calls and loops. *WARNING: If an application writes to ROM address space while this setting is enabled, it could cause problems for that application and for every application run after that.*

**Exclusive Mouse Access**

**Function:** This setting allows Virtual DOS Sessions to run applications that maintain their own mouse pointers. Some DOS applications manage their own mouse positions and movements; in many cases, the application's values for mouse sensitivity or double-speed threshold are different from those of Presentation Manager. As a result, a PM mouse pointer can be outside the Virtual DOS Session window while the application pointer is somewhere in the window that is not receiving any mouse events.

**Advantages:** The user forces the physical mouse driver to send its events directly to the virtual mouse driver without going through PM. Only one mouse pointer appears when the particular Virtual DOS Session window has the focus.

**Default:** Off.

**Settable:** At any time from the Virtual DOS Session system menu, by selecting Properties and turning on the Exclusive Mouse Access setting. However, this only marks the Virtual DOS Session window; it does not actually activate the setting. To activate it, the user must press a mouse button within the Virtual DOS Session window. The PM pointer disappears, leaving only the application pointer. To regain the PM pointer, the user must press any of the hot keys (Alt, Ctrl+Esc, Shift+Esc).

**Example:** When a user runs an application such as Microsoft Windows, PC PaintBrush<sup>®</sup>, or Flight Simulator, two mouse pointers appear in the window and may move out of sync with one another. This setting avoids such situations.

**Video: Fast Paste**

**Function:** Simulates user input using a faster mechanism.

**Advantages:** Improves the speed of paste operations made from the clipboard to a DOS application.

**Drawbacks:** Does not work with all applications; in particular, some applications that monitor keyboard interrupts directly may experience errors.

**Default:** Off.

**Settable:** At any time. This facilitates easy experimentation by the user.

**Example:** Pasting into the DOS command prompt, or into any application using DOS console I/O functions, usually works. However, the M editor by Microsoft, and its successor, Programmer's Workbench, can fail when using fast pasting because they rebuffer keystrokes in an internal buffer that can overflow.

Figure 2. Hardware Environment Settings (continued)

### Idle Seconds Allowed

**Function:** When a program appears to be waiting for input, OS/2 gives it less time and gives preference to other programs that are doing useful work. Some programs periodically appear to wait for input, but then deliberately continue after a time. This setting disables the "idle detection" function for a period after useful work has been detected, and makes allowance for programs that resume after waiting.

**Advantages:** If a program appears to run slowly when there is an option for the user to provide input, this value should be increased so that OS/2 waits longer before reducing the execution time allocation for this program.

**Default:** This value is in seconds, and the default is no idle time allowed.

**Settable:** While the program is running, this setting can be changed to tune it to the proper value.

**Example:** A game may pause to wait for the user to make a choice, but then continue if the user does not react.

### Idle Sensitivity

**Function:** The idle sensitivity level sets a threshold for judging when applications are idle. The value is the percentage of the maximum possible polling rate that the application can perform. The polling rate is a factor of processor speed, application priorities, and the number of tasks that are active in the system. If an application polls at a rate higher than this value, it is considered to be idle.

Idle detection is a "best guess" of what the program is doing. It could be that the program is polling at a very high rate, but is still doing useful work between checking. It may be that the application checks at a slow rate, but still is doing nothing but waiting. The idle sensitivity threshold allows adjustment of the threshold for a particular application.

DOS programs often "poll" for input when they are waiting for a user response. For instance, a program may wait for a response by repeatedly checking to see if the user has hit a key. In a multitasking environment such as OS/2, this wastes time when other programs could be running instead. The operating system detects idle programs by looking for a high rate of polling for input. When programs are judged to be waiting for input, they are given less time to run.

For example, if idle sensitivity is set to 75%, then an application that repeatedly checks whether input is available would have to do this checking at more than 75% of the maximum possible polling rate before it would be judged idle.

For some communication applications (such as ProComm Plus<sup>®</sup>) and full-screen video applications, it is better to set this value low, perhaps at 30 rather than 75. Also, tune the TIMESLICE parameter in CONFIG.SYS if system performance is still noticeably slow. Start with TIMESLICE=64, 128 and tune from there.

**Advantages:** If an application receives input while running but seems to run slower than expected, the idle sensitivity should be set to a higher value. This lets the application poll at a higher rate without being judged idle. Setting the level to 100 turns idle detection off altogether, and the application will be allowed to poll for input as often as it likes.

If an application is waiting for input and other applications do not appear to be running, the idle sensitivity should be adjusted downward. This lowers the threshold for proclaiming the application idle.

**Default:** 75%.

**Settable:** The setting can be changed while the program is running to tune it to the proper value.

### IOPL Lock

**Function:** It enables or disables trapping instructions that affect the interrupt flag.

**Advantages:** It resolves problems that can be encountered with certain copy-protected or timing-sensitive applications.

**Drawbacks:** It can increase interrupt latency to a potentially unusable level.

**Default:** Off.

**Settable:** At Virtual DOS Session creation time only.

**Example:** F15, a copy-protected flight simulation game, will not run without this feature.

Figure 2. Hardware Environment Settings (continued)

**Keyboard: Buffer Extension**

**Function:** It increases a Virtual DOS Session's keyboard type-ahead buffer size.

**Advantages:** It provides greater keystroke buffering, consistent with the level available in Virtual Input/Output (VIO) windows. Note that Ctrl-Break flushes the entire buffer, just as it does with the standard buffer.

**Drawbacks:** Applications that bypass the ROM BIOS input buffer or INT 16h may not benefit from this feature. There also is a small amount of additional memory overhead for every Virtual DOS Session.

**Default:** On. Most applications will benefit, and those that do not should not be adversely affected.

**Settable:** At any time. This facilitates easy experimentation by the user in the rare event that a problem arises.

**Timer Hardware Access**

**Function:** This setting allows direct access to the 8253 timer ports.

**Advantages:** Certain timing-critical applications will not run, or will run much slower, if accesses to timer ports are trapped and virtualized (that is, if applications access a virtual timer instead of the hardware timer). In addition, the values they read do not accurately reflect the amount of time that has passed, because they do not take trapping overhead into account. Enabling this setting allows certain timing-dependent code to run more effectively.

**Drawbacks:** Applications that change the divisor before this setting is enabled, then read the timer ports after the setting has been enabled, may not function properly. If the setting is enabled first, the Virtual DOS Session cannot correctly detect changes to the divisor, and the simulated interrupt frequency will be incorrect. Multiple applications using this setting can interfere with one another.

**Default:** Off. Most applications will operate normally with a virtual timer.

**Settable:** At any time. It is useful to alter this setting dynamically and to watch for changes in application performance.

**Example:** The ROM on some computers implements very brief delays by polling the timer ports. These delays become unacceptably long unless direct timer port access is allowed.

Figure 2. Hardware Environment Settings

**DOS Device Drivers**

**Function:** This setting can be used to add or modify information about DOS device drivers, in addition to the information specified in CONFIG.SYS. Do not specify device drivers that applications do not use, because each loaded driver takes space from DOS applications that run in this session.

**Default:** When this setting is selected, a list is displayed that contains information about each DOS device driver specified in CONFIG.SYS. This information consists of the path and file name of each DOS device driver and its current parameters, if applicable. An example of this information is as follows:

```
c:\os2\mdos\ansi.sys
```

The user can:

- Type (as the value of this setting) the name of a DOS device driver to add it. Typing should begin on a new line.
- Delete all the information about a device driver to remove it.
- Type additions, changes, or deletions to a value.

**Settable:** At Virtual DOS Session creation only.

**DOS Memory Size**

**Function:** This setting specifies the DOS memory size in KB. This is the amount of memory available to DOS applications.

**Advantages:** The virtual video device driver uses this setting on certain video adapters to set more than 640 KB.

**Drawbacks:** This setting is insignificant to most users because there is no reason to specify less than 640 KB.

Figure 3. DOS Environment Settings (continued)

<p><b>Default:</b> 640 KB.</p> <p><b>Settable:</b> At Virtual DOS Session creation only.</p>
<p><b>Last Drive</b></p> <p><b>Function:</b> This setting specifies the highest disk drive letter accessible to the DOS session.</p> <p><b>Advantages:</b> Restricting the highest disk drive letter saves a small amount of standard memory in the DOS session.</p> <p><b>Drawbacks:</b> The DOS session is restricted to drive letters less than or equal to the last drive specified.</p> <p><b>Default:</b> Z.</p> <p><b>Settable:</b> At Virtual DOS Session creation only.</p>
<p><b>Memory: Load DOS High</b></p> <p><b>Function:</b> This setting specifies that DOS be loaded in the High Memory Area (HMA) if set on.</p> <p><b>Advantages:</b> It adds more than 40 KB to the standard memory area available to DOS application programs.</p> <p><b>Drawbacks:</b> Some DOS programs require that DOS be loaded low.</p> <p><b>Default:</b> On.</p> <p><b>Settable:</b> At Virtual DOS Session creation only.</p>

Figure 3. DOS Environment Settings

<p><b>Memory: DPMI</b></p> <p><b>Function:</b> This setting defines the maximum amount of DPMI memory available to all applications running in this DOS session. The field for this setting contains values expressed in 1 MB intervals ranging from 0 to 512 MB. Select 0 if the applications to be run in this session do not require DPMI.</p> <p><b>Advantages:</b> DPMI allows applications to run in protected mode, which allows application programs to address more than the 1 MB of memory available in real mode, the standard mode used by DOS applications. DPMI is required to run Windows applications in their standard mode. The application's documentation should indicate if the application can take advantage of DPMI memory.</p> <p><b>Drawbacks:</b> Specifying DPMI memory for a DOS session incurs only a small amount of overhead if the memory is not used. Specifying excessive values of DPMI for many DOS sessions could result in exhaustion of system swap space.</p> <p><b>Default:</b> 3 (MB).</p> <p><b>Settable:</b> At Virtual DOS Session creation time only.</p>
<p><b>Memory: EMS Frame Location</b></p> <p><b>Function:</b> The Lotus<sup>®</sup>-Intel-Microsoft Extended Memory Specification (LIM EMS) uses a 64 KB address region, called an EMS page frame, through which programs access expanded memory. This enables programs to use more than 640 KB of memory.</p> <p><b>Advantages:</b> If there is a problem running a program that uses both a hardware device and LIM EMS expanded memory, it may be due to conflicting use of addresses by LIM EMS and the hardware device. If so, the EMS Frame Location setting should be changed to specify the extra address region used by EMS as 0. If the problem persists, the EMS Frame Location setting can be used to select a 64 KB region that does not conflict with hardware.</p> <p>The frame location can be selected from a list of choices or the EMS frame can be omitted for programs that do not require a frame. Also, the DOS memory size setting can be reduced and the frame placed below 640 KB.</p> <p><b>Drawbacks:</b> Rather than using this setting, the best solution for problems due to hardware address conflicts is to use the other Exclude Regions and Include Regions settings to specify the addresses that the hardware uses. Care must be taken when using these settings, because detailed knowledge of your computer's hardware address usage is required.</p> <p><b>Default:</b> The default setting, AUTO, leads to correct choices of LIM EMS addresses. Most users will never need to change this setting.</p>

Figure 4. Memory Extender Settings (continued)



**Settable:** At Virtual DOS Session creation time only.

**Example:** In some cases, the default choice may conflict with addresses used by the computer's hardware. This can happen only for devices that are not supported by a virtual device driver.

#### **Memory: EMS High OS Map Region**

**Function:** In addition to the EMS page frame, some programs can use additional addresses to access expanded memory. This setting allows advanced users to adjust the size of the additional EMS region.

**Advantages:** Advanced users can use other Exclude Regions and Include Regions settings to specify the addresses used by devices that do not have virtual device drivers, and can then set the size of the high OS region appropriately for their programs. Care must be taken to ensure that the hardware addresses do not conflict with other addresses being used within OS/2.

**Default:** The value set is the size of the region in KB. The default is 32 (KB).

**Settable:** At Virtual DOS Session creation time only.

#### **Memory: EMS Low OS Map Region**

**Function:** Some programs can use remappable conventional (RAM) memory. This setting allows advanced users to set the size of the remappable conventional memory available in a Virtual DOS Session.

**Default:** The value set is the size of the region in KB. The default is 384 (KB).

**Settable:** At Virtual DOS Session creation time only.

#### **Memory: EMS Memory Size**

**Function:** This setting controls the amount of EMS memory available to a Virtual DOS Session.

**Advantages:** The user can set this to a higher value for running programs that require a large amount of EMS. Other programs do not use EMS at all. To disable EMS support in such cases, the size can be set to 0 for that Virtual DOS Session. Programs generally state whether they use EMS on the product box or in their manuals.

**Default:** The value set is the size of the region in KB. The default size is 4 (KB).

**Settable:** At Virtual DOS Session creation time only.

**Example:** If a spreadsheet runs out of memory, the amount of EMS can be increased and the Virtual DOS Session restarted.

#### **Exclude Regions**

**Function:** This setting specifies the address ranges that should be protected from use by EMS or XMS and from direct access by applications. This setting is intended for experienced users who understand the hardware address ranges that are used in OS/2. If an incorrect setting is specified, unpredictable results can occur.

**Advantages:** This setting restricts the use of EMS and XMS in certain ranges in the region between Real Memory Size (RMSIZE) and 1 MB. It also protects these ranges from being accessed by user applications by portraying ROM in those address ranges.

**Drawbacks:** Some hardware adapters stop functioning if their addresses are accessed in random fashion. If these ranges are defined excessively, they will adversely impact the function and performance of EMS and XMS services.

**Default:** By default, this setting is void (that is, nothing follows the equal sign). Each address is specified in hexadecimal; if no range is specified, the length taken is a page (4 KB).

**Settable:** At Virtual DOS Session creation time only.

#### **Include Regions**

**Function:** This setting specifies some address ranges between RMSIZE and 1 MB for use by EMS and XMS.

**Advantages:** If the user knows that there is a hardware adapter in this range that is not going to be used by a particular Virtual DOS Session, then the address range used by this adapter should be made available to EMS and XMS. This will improve the performance of EMS and XMS services. Only advanced users who know the addresses used by an adapter card should use this setting.

**Default:** By default, this setting is void.

**Settable:** At Virtual DOS Session creation time only.

**Figure 4. Memory Extender Settings (continued)**

**Memory: DOS Owns Upper Memory Blocks (UMBs)**

**Function:** The use of Upper Memory Block (UMB) areas allows certain types of DOS code, such as Terminate-and-Stay-Resident (TSR) programs and DOS device drivers, to be loaded into memory addresses between 640 KB and 1,024 KB. This frees memory in the 0 to 640 KB range for application use.

The DOSEM kernel owns and manages all UMB areas. Code is then loaded into UMB areas by the DEVICEHIGH and LOADHIGH commands. However, if an application program needs to manage UMB areas, the DOS emulation kernel must relinquish ownership and management of this memory. This can be done for individual Virtual DOS Sessions that turn off the DOS-owned UMB setting.

DOS ownership of UMB areas can be enabled or disabled for all Virtual DOS Sessions by including either DOS=UMB or DOS=NOUMB in CONFIG.SYS.

**Default:** By default, DOS=UMB (DOS owns UMB) is in effect (On).

**Settable:** At Virtual DOS Session creation only.

**Memory: (Number of) XMS Handles**

**Function:** This setting specifies the number of XMS Extended Memory Block (EMB) handles. A handle is used with each EMB. This number is required because XMS preallocates all the handle space to be compatible with XMS specifications. This setting should be used only if an application uses a large number of handles.

**Advantages:** This setting restricts the number of block handles, thereby reducing memory consumption.

**Drawbacks:** Specifying a large number of handles increases memory consumption and adversely impacts system performance.

**Default:** The default value is 32.

**Settable:** At Virtual DOS Session creation only.

**Memory: XMS Memory Limit**

**Function:** This setting specifies the global and individual Virtual DOS Session XMS memory limits. Use this setting under the same guidelines as described in the preceding section on XMS Handles. The global limit is the overall maximum XMS memory consumption, and the individual Virtual DOS Session limit is the maximum allowed for each session.

**Drawbacks:** Specifying a large number may adversely affect system performance.

**Default:** The default value is 2048 (KB) for an individual Virtual DOS Session.

**Settable:** At Virtual DOS Session creation only.

**Example:** The setting 8192,4096 specifies 8 MB for the overall limit and 4 MB for each Virtual DOS Session.

**Memory: XMS Minimum Usage of High Memory Area (HMA)**

**Function:** This setting specifies the minimum HMA memory request allowed. This setting lets the user fine-tune the XMS. HMA is slightly less than 64 KB in size. Only one request at a time can be fulfilled from this area, and only a real-mode application can use this memory as conventional memory.

**Advantages:** If a TSR takes a very small allocation, it will waste this area for other applications. In such cases, a limit can be specified.

**Default:** The default value is zero, which means that all requests will be allowed.

**Settable:** At Virtual DOS Session creation only.

**Example:** The 2048 sets a limit of 2 KB.

**XMS UMB Deactivation**

**Function:** UMBs are the regions anywhere between RMSIZE and 1 MB where the memory is available to XMS. All the regions specified in other Include Regions are available here.

**Advantages:** When this setting is enabled, XMS waits for the first UMB request before allocating all UMBs. This is done to give UMBs better hardware support.

**Default:** By default, UMBs are not supported to minimize conflict with adapters; however, if an application depends on UMBs, this setting should be enabled.

**Settable:** At Virtual DOS Session creation only.

Figure 4. Memory Extender Settings (continued)

**FCB Count**

**Function:** This setting specifies the maximum number of File Control Blocks (FCBs) that can be opened. This setting has no effect unless a file-sharing module is loaded.

**Advantages:** In a networking environment, the performance of a Virtual DOS Session can be severely impacted if a large number of FCBs are active. Through this setting, the user can specify a limit to such file openings. If the open count crosses this limit, DOS closes the least recently used FCB.

**Default:** The default value is 16.

**Settable:** At Virtual DOS Session creation only.

**FCB Count, No Close**

**Function:** This setting specifies the number of FCBs that will be protected against automatic closure.

**Advantages:** If this setting is specified as a number  $n$ , the first  $n$  files are protected against automatic closure as described in the previous setting.

**Default:** The default value is 8.

**Settable:** At Virtual DOS Session creation only.

**Figure 5. File Operation Settings**

**Video: Window Refresh**

**Function:** This setting adjusts the window update frequency (in seconds) for a given Virtual DOS Session.

**Advantages:** For applications (particularly graphics) that write frequently to video memory, this value can be increased to reduce the time spent updating the window and to provide more processor time for the application. This setting has no effect on updates based on other events such as keyboard input or synchronous scrolling operations.

**Drawbacks:** A large refresh period can make an application unusable or very hard to use.

**Default:** The default value is 0.1 (seconds). This value has been found to yield the best overall performance.

**Settable:** At any time, in increments of 0.1 seconds. The range is from 0.1 to 60.0 seconds.

**Example:** This setting affects normal TTY-style output. Compare a DIR or TYPE operation before and after altering this setting.

**Video: Window Scroll Synchronization**

**Function:** This setting skips window updates on individual scroll operations.

**Advantages:** Scrolling in a window is several times faster. Use this setting in conjunction with Window Refresh to obtain a desirable refresh rate.

**Drawbacks:** Scrolling is not as smooth as full-screen operation.

**Default:** On. In most cases, smooth scrolling will be more important to users than very fast scrolling.

**Settable:** At any time. This permits experimentation.

**Example:** This setting affects normal TTY-style output. Compare a DIR or TYPE operation before and after altering this setting.

**Figure 6. Windowing Settings**

to close and release all resources when they are no longer needed. This includes files, windows, dialogs, and memory. Remember that with the flat memory model, all variables are 32 bits in length. Take advantage of the 32 bits for flags and addresses. Also, be sure to align the code and data on 32-bit boundaries to optimize memory accesses.

*Ron Morrill joined IBM in New York as a systems engineer on the City College team in 1968. Since that time he has supported TSO, APL, CP67, and other time-sharing systems. He has been a system support representative for HONE and a systems engineering manager supporting Texas Instruments in Dallas, Texas. He is currently a senior marketing support representative supporting OS/2.*

*Ron Cadima is an advisory programmer in the OS/2 Performance Support group in the IBM Personal Systems Programming Center in Boca Raton, Florida. In his 24 years with IBM, Ron has worked on operating systems development, production and process control applications, communications with point-of-sale systems, and OS/2 system performance. He was the lead performance analyst for OS/2 1.3, and he currently works with customers, independent software vendors, and others in designing OS/2 applications and tuning OS/2 systems for performance.*

# Cleaner Installation of Applications Under OS/2

**Claude Goffin**  
**IBM Corporation**  
**La Hulpe, Belgium**

*Installing, customizing, and maintaining OS/2 applications can be complex and time-consuming, because some application installation procedures add new directories to the hard disk and modify the operating system's CONFIG.SYS file. This article presents some tips for installing and accessing applications more cleanly, without changing the CONFIG.SYS file each time a new application is added.*

Every session running under OS/2 has its own environment, and each session's environment can be modified. The *environment* of an OS/2 session is a special area in storage that contains information specific to that OS/2 session. That information consists of the subjects discussed in this article: system variables, user variables, the current drive, and the current directory for every accessed drive. *Accessed drives* consist of all the physical and logical drives defined in the OS/2 installation; a local virtual disk, defined by the parameter `DEVICE=C:\OS2\VDISK.SYS` in `CONFIG.SYS`; or a remote virtual disk residing on a Local Area Network (LAN) server. The entries in the `CONFIG.SYS` file define the default environment of all OS/2 sessions running concurrently.

Most application vendors recommend modifying `CONFIG.SYS` to include their specific requirements. The advantage of modifying `CONFIG.SYS` is that an application can then be used by any session. But there also are adverse effects. The chain of directories to be searched for a specific file is long. The `CONFIG.SYS` file becomes unwieldy. Worse, even after users remove an

application from their computers, they rarely remove the command lines in the `CONFIG.SYS` file that were added when the application was installed. Finally, application vendors have many different methods of modifying `CONFIG.SYS`, and these differences confuse users. These situations can lead to difficulties in customizing, maintaining, and debugging a system. It also can cause loss of data files.

Sometimes a new program does not meet expectations, and it is removed from the system. Because of this possibility, it is strongly advised that you make a copy of the existing `CONFIG.SYS` file before installing any new program. Some programs attempt to do this for you by renaming the existing `CONFIG.SYS` file to `CONFIG.BAK` or a similar name. However, this can lead to a conflict if you already have a `CONFIG.BAK` and you are now installing a program that wants to create a different `CONFIG.BAK`. Here is a suggestion: Each time you install a new application, save the most recent `CONFIG.SYS` file under the name `CONFIG.NNN` (where `NNN` can be a Julian date or a simple sequential number), and store `CONFIG.NNN` in a directory of backups — such as `D:\BACKUP\CONFIG.NNN`.

Because of all the drawbacks of modifying `CONFIG.SYS`, an application installation strategy can minimize changes to `CONFIG.SYS` and make installation and maintenance significantly easier. The installation strategy discussed in this article takes advantage of OS/2. The purposes of this article are to explain this approach and to help users and technical support personnel develop their own installation strategies.

The next four sections discuss the concepts of the OS/2 session environment, user and system variables, system search sequences, and Dynamic Link Libraries (DLL). If you are familiar with these concepts, you may wish to skip to the section "General Guidelines."

## User and System Variables

Because OS/2 is a multitasking operating system, it allows concurrent occurrences of multiple OS/2 sessions in which batch files (`.CMD`) or applications are executed. It is important to keep in mind that every OS/2 session has its own environment.

System variables are defined by using `SET` commands in `CONFIG.SYS`. This article covers the following system variables:

- `PATH` specifies the search path for executable files (`.CMD`, `.COM`, `.EXE`).
- `DPATH` specifies the search path for data files.
- `HELP` specifies the search path for help files (`.HLP`).
- `BOOKSHELF` specifies the search path for information files (`.INF`).

User variables are created and defined by using `SET` commands that are issued at the OS/2 system prompt or in a batch file. User variables are the replaceable parameters in batch (`.CMD`) files. Here is an example of their usage.

Suppose CHECK.EXE is a program that checks and corrects the file D:\WORKING.DAT. This file is a temporary workspace into which other "original" files are copied, then CHECKed, then copied back.

Assume one of these original files is D:\USER\WORK\BALANCE.DAT. To CHECK this file in an OS/2 session, you might use these commands:

```
COPY D:\USER\WORK\BALANCE.DAT
  D:\WORKING.DAT
CHECK
COPY D:\WORKING.DAT
  D:\USER\WORK\BALANCE.DAT
```

To CHECK another original file, it is necessary to use another set of these commands with a different file name substituted for D:\USER\WORK\BALANCE.DAT. Clearly, this can get tedious if you want to CHECK many files.

There is an easier way to accomplish this. Use the SET command to define and fill a user variable with the name of the original file. In the above example, the commands for defining and filling a user variable MYDATA are as follows:

```
SET MYDATA=D:\USER\WORK\BALANCE.DAT
COPY %MYDATA% D:\WORKING.DAT
CHECK
COPY D:\WORKING.DAT %MYDATA%
```

The file name D:\USER\WORK\BALANCE.DAT is substituted for the user variable MYDATA. In the COPY statements, the percent signs on both sides of the name MYDATA indicate that MYDATA is a user variable that is replaced by an actual value (in this case, a file name) before execution.

Next, create a batch file D0.COM that contains the last three commands shown above:

```
COPY %MYDATA% D:\WORKING.DAT
CHECK
COPY D:\WORKING.DAT %MYDATA%
```

The combination of the user variable MYDATA and the batch file D0.COM can now be used to CHECK several files:

```
SET MYDATA=D:\USER\WORK\BALANCE.DAT
D0
SET MYDATA=D:\USER\WORK\ASSETS.DAT
D0
SET MYDATA=D:\USER\WORK\BILLS.DAT
D0
.
```

The SET command also can be used in a batch file. In this case, however, you must edit that batch file each time you want to modify the value of one user variable.

System variables also can be used as replaceable parameters. The system variables %PATH%, %DPATH%, %HELP%, and %BOOKSHELF% can be used in commands and batch files. You can then assign actual values to these variables with the SET command; for example:

```
SET PATH=D:\UTIL
```

The current values of all user and system variables in an individual OS/2 session can be seen any time by typing the command SET.

## Changing System Variables

Within a single OS/2 session, the values assigned to user and system variables in SET statements are valid only for that session. No other sessions are affected. Often, however, the system variables must have default values in all concurrent OS/2 sessions. This is where the CONFIG.SYS file becomes important. Every time the system is powered on, OS/2 searches the root directory of the drive where it started (usually the C: drive) for the CONFIG.SYS file. The CONFIG.SYS file is a batch file that OS/2 interprets to assign values to variables and set up the system according to the devices that are attached. OS/2 performs these proce-

dures before it permits individual sessions to start.

In OS/2, one purpose of CONFIG.SYS is to provide every OS/2 session with a default set of variables. The default variables for all sessions are defined by using SET commands in the CONFIG.SYS file.

At system boot time, CONFIG.SYS defines a default set of system variables, and eventually user variables. To make a permanent change, you must edit the CONFIG.SYS file, then reboot the system. To change a variable temporarily in an individual OS/2 session, either rewrite the SET statement or add new parameters to it. Two examples follow.

To replace the path defined in CONFIG.SYS with a new one:

```
SET PATH=D:\USER\APPS;
```

To add this same new entry to the beginning of the current path:

```
SET PATH=D:\USER\APPS;%PATH%
```

*Note:* You can use the commands PATH and DPATH instead of SET either on the command line or in a batch file, but not in CONFIG.SYS. However, for simplicity, this article uses only the SET command.

## Current Directories and Search Sequences

Each OS/2 session has its own current directories, as shown in the following example. Suppose you are using two drives, C: and D:. In a single OS/2 session, you issue the following commands:

```
C:
CD C:\DIRC
D:
CD D:\DIRD
PROG
```

Notice that the second line includes the full path (C:\DIRC), not just

\DIRC or DIRC. (The OS/2 2.0 *Command Reference* manual has examples that use \DIRC or DIRC. These examples are correct, but they do not consider the pitfall explained here.) It is safer to include the full path, especially when you are running a batch file. For example, if you had previously issued the command CD \WORK, then the command CD DIRC would make C:\WORK\DIRC (if it exists) the new current directory. If C:\WORK\DIRC does not exist, the session will abruptly stop when it tries to find this non-existent file later.

At the point of invoking PROG:

- The current directory of the C: drive is C:\DIRC.
- The current directory of the D: drive is D:\DIRD.
- The current directory of the session is also D:\DIRD.

PROG is the name of the executable file (.EXE, .COM, or .CMD) to be run. When OS/2 searches for executable files to run in an individual session, it checks the last SET PATH= statement issued during that session. If no SET PATH= statement has been issued, OS/2 then checks the SET PATH= statement in CONFIG.SYS.

In OS/2, the dot (.) can be used in SET PATH= statements to indicate the current directory. For example, suppose that in the OS/2 session, you issue the statements:

```
SET PATH=.\UTIL;C:.;%PATH%
C:
CD C:\DIRC
D:
CD D:\DIRD
PROG
```

OS/2 then searches for the executable file PROG in the following order:

1. Current directory of the session (D:\DIRD) (If the needed file is not found in the current directory, the search is done as defined in the

PATH system environment variable.)

2. Subdirectory \UTIL of the current directory, if it exists (D:\DIRD\UTIL)
3. Current directory of drive C: (C:\DIRC)
4. Directories in the previous value of the PATH variable for that session (This is generally the value that was set in the SET PATH= statement in CONFIG.SYS.)

Using the dot also is valid in the following statements:

- SET DPATH= establishes the search path for data files.
- SET HELP= establishes the search path for help files (.HLP).
- SET BOOKSHELF= establishes the search path for documentation files (.INF).

## Dynamic Link Libraries

All running OS/2 sessions can share Dynamic Link Libraries (.DLL) files. The .DLL files are loaded into memory when needed. At runtime, an application may need routines stored in .DLL libraries, so the application must know the path for finding the .DLL files. This is the reason for the LIBPATH= command in CONFIG.SYS. (The LIBPATH is not part of the environment of an individual OS/2 session; it applies to all concurrent OS/2 sessions, so it cannot be defined using a SET command.) DLLs common to multiple applications should be put into a single directory that is specified in the LIBPATH= statement.

There is no naming convention for DLLs. Eventually, identical names for .DLL files will appear in different applications. It may be helpful to group the DLLs specific to one application with the other files for that application. That is what many program installation procedures do. Although

it is not necessary, the installation procedures also want to update the LIBPATH= statement in CONFIG.SYS to point to their particular applications. Doing this has some negative effects:

- The LIBPATH= statement becomes cluttered – too long and difficult to read.
- The search sequence for the needed file is long.
- Identical DLL names in different applications can cause a search to locate the wrong .DLL file.
- This arrangement is not well suited to a LAN environment where applications reside on a server.
- When one wants to remove an application, and therefore the DLLs belonging to it, there is almost no way to know which DLLs to remove.

When an OS/2 session searches for DLL files, it does not search the current directory of that OS/2 session first. Instead, the search sequence is the one specified in the LIBPATH= statement; this search sequence is the same for all concurrent OS/2 sessions.

Therefore, when program installation procedures modify – in fact, lengthen – the LIBPATH= statement in CONFIG.SYS, each OS/2 session has to search unnecessarily through all .DLL files, not just the ones it needs.

Fortunately, the dot feature provides an alternative for searching for DLLs. Here is how it works. Assume the LIBPATH= statement in the CONFIG.SYS file is as follows:

```
LIBPATH=.;C:\OS2\DLL;
C:\MUGLIB\DLL; ...
```

In this example, the dot follows the equal sign. This dot represents the current directory of an individual OS/2 session. Therefore, although the

value of LIBPATH will be the same for every OS/2 session, the value that replaces the dot will be different for each session. Here is a practical example of using the dot instead of modifying the LIBPATH= statement in CONFIG.SYS.

Suppose you are installing a program called ALONE, and its installation process modifies the LIBPATH= statement above to look like this (all in one line):

```
LIBPATH=C:\ALONE;C:\OS2\DLL;C:\MUGLIB\DLL; (rest of the statement)
```

This permanently changes the LIBPATH. Every OS/2 session that needs DLL files will look for them in the ALONE directory first, even though the DLL files in the ALONE directory apply only to the session that is running the ALONE program.

A better alternative is to start the LIBPATH= statement with a dot, as shown above. Then issue the following statements to start the ALONE program:

```
C:
CD C:\ALONE
ALONE
```

The term C:\ALONE will be substituted where the dot is. As a result, *for the ALONE session only*, the LIBPATH will behave as though it begins with C:\ALONE.

Now suppose that in another OS/2 session you want to execute a program called NEXTPROG, and that the program and its DLLs reside in D:\NEXTPROG. The following statements will execute NEXTPROG, and the search for DLLs will begin in the D:\NEXTPROG directory *for this session only*:

```
D:
CD D:\NEXTPROG
NEXTPROG
```

## General Guidelines

It is now time to use the concepts learned thus far. The rest of this article gives guidelines for establishing and adhering to an installation and maintenance strategy that facilitates application installation, system maintenance and backup, and saves debugging time.

### OS/2 System Files

The C: drive, whether physical or logical, should be reserved for OS/2 system files. When all OS/2 system files are in one place and are isolated, it becomes much easier to upgrade some or all of those files. They can be upgraded from diskettes, internal or external tape drives, a PS/2-to-PS/2 connection, or through a connection to a LAN server or to a host computer. Before upgrading, be sure to back up CONFIG.SYS, all .INI files, and the STARTUP.COM file (if this file is installed).

There is another big advantage of putting all system files onto one drive. If there is more than one operating system on your computer, the files of each operating system can go onto their own drive. You can then run the applications (which are on a separate

drive) from each of those operating systems.

### Application Files

With few exceptions, each application should have its own directory or subdirectory for placing all files from the application package. Again, the obvious advantage is that it then becomes simple to install a new version of that application. There are some exceptions:

- Some tools consist only of DLLs. A well-known example is REXX extensions, which can be used by several programs. Those DLLs are good candidates for placing into a common DLL directory referred to in the LIBPATH= statement.
- Some tools consist of only one .EXE file. These tools are used in any session through the command line or in another .CMD. They are good candidates to place into a common utility directory referred to in the SET PATH= statement.

### Personal Files

Personal files consist of data files and control files. *Data files* are texts, spreadsheets, and so on. They can be grouped according to the applications



that access them; most data files are accessed by only one application. *Control files* usually contain parameters that can be customized for an application. These files are typically named Profile (.PRO) or Configuration (.CGF); they also can be customizable batch files. *Personal files* contain information such as the name of the directory for storing temporary and final results; autosave intervals; alarms; names of diaries; programmed keystrokes; color and sound preferences; and personal dictionaries for spell-checking and proofreading.

As with system and application files, it is best to put related personal files

into their own separate directory or subdirectory.

Most application packages contain default control files that can be changed based on needs. To do this, copy each application's default control files into a personal directory or subdirectory so you can edit those personal control files. Then, if you install a new version of the application, the personal control files will not be overwritten because they are in another directory.

### CONFIG.SYS

Statements in CONFIG.SYS should be as short and stable as possible. Apply-

ing the strategies in this article should remove the need to modify CONFIG.SYS.

### Starting Applications

To set up the environment of an OS/2 session in which an application will run, use batch (.CMD) files. Every application can be started with a .CMD file. This starting batch file can be executed from an OS/2 system prompt or by double-clicking on the "Starting Application" icon. When creating the corresponding program object, this .CMD file is the one to specify in the path and file name of the Program tab in the Settings notebook. The starting batch file also is a good candidate to put into the common utility directory referred to in the SET PATH= statement in CONFIG.SYS.

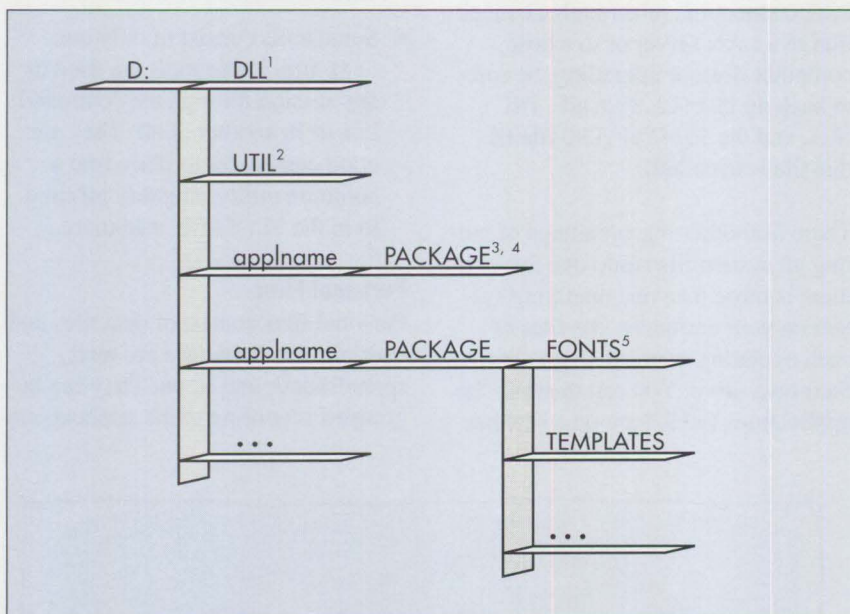
The .CMD file that starts the application has the following structure:

1. Define current directories for the accessed drives and for the OS/2 session
2. SET the variables for the application (if any)
3. appl.exe
4. EXIT

The starting batch file sets up the environment for the desired application, and then starts that application.

App.exe is a generic term for an executable file that runs an application; for example, the actual executable file might be D:\ALONE\ALONE.EXE (it is good practice to use the full path).

After double-clicking on the "Starting Application" icon, it appears gray (or cross-hatched) indicating that the application has been started. In the course of your work, you will have minimized the application window. Double-clicking again on the "Start-



Notes:

<sup>1</sup> D:\DLL stores DLLs common to all applications.

<sup>2</sup> D:\UTIL stores all executable files of tools (except those that came with the OS/2 system). It also is the place to store all the starting batch files that are created.

<sup>3</sup> D:\applname\PACKAGE stores all original files that came with the applname application (.EXE, .DLL, .INF, .HLP, and so on).

<sup>4</sup> D:\applname stores all personal files related to the applname application. It is not always obvious which files to copy from D:\applname\PACKAGE to D:\applname. When you can modify an application's settings, it is good to check which files have changed, to copy those files into the D:\applname directory, and then to restore the original files from the application package.

<sup>5</sup> Some applications also create subdirectories. Generally, these subdirectories do not affect the installation strategy.

Figure 1. Directory Tree for a Single Drive



ing Application" icon restores the application window to its normal size.

An OS/2 window icon related to the starting batch file is created in the minimized Window Viewer. This is useful while testing, but you may keep it from being created by selecting "Hide Window" in the Window tab of the Settings notebook.

Not all applications require a starting batch file because the options available in the Settings notebook are sufficient. However, there is little or no inconvenience in using this approach, and experience shows that users who adopt this approach never encounter problems.

### Proposed Strategies

It is now time to apply the guidelines to specific strategies. This section dis-

cusses installation and maintenance strategies for two scenarios: running applications using a single drive and running applications using two drives. Each scenario includes a suggested directory tree layout, a modified starting batch file, and a modified CONFIG.SYS file.

#### Running Applications on a Single Drive

Running applications on one drive (D:) in addition to the operating system drive (C:) makes disk space management easy and works well in most cases. The suggested directory tree is shown in Figure 1.

For each application `appName`, create the starting batch file `D:\UTIL\appName.COM`, as shown in Figure 2.

```
D:
CD D:\appName
SET the variables for the application (if any)
D:\appName\PACKAGE\APPLEXE (assuming APPLEXE is the program to start)
EXIT
```

Figure 2. Starting Batch File for a Single Drive

```
LIBPATH=.;.\PACKAGE;D:\DLL;C:\OS2\DLL;...1,2
SET PATH=.\PACKAGE;D:\UTIL;C:\OS2\;...3
SET DPATH=.\PACKAGE;C:\OS2;...3
SET HELP=.\PACKAGE;C:\OS2\HELP;
SET BOOKSHELF=.\PACKAGE;C:\OS2\BOOK;
```

Notes:

<sup>1</sup> The `.;` at the beginning of the `LIBPATH` is not needed, because the DLLs are in `D:\appName\PACKAGE` rather than in the current directory (`D:\appName`) of the session.

However, you should not remove the `.;` that OS/2 2.0 has included, because some applications may run from their own directories, or because you may be testing a new application for which the starting batch file has not yet been created.

<sup>2</sup> For this OS/2 session, `D:\appName\PACKAGE` will be searched for DLLs first. There will be no unnecessary search for DLLs in other directories, and minimal risk of encountering DLLs with identical names.

<sup>3</sup> For the `PATH` and `DPATH`, the current directory will be searched first, before `D:\appName\PACKAGE`. If the control files were copied into `D:\appName` (the current directory), the customized control files are accessed first, and the default control files (which have the same names) in `D:\appName\PACKAGE` are not accessed.

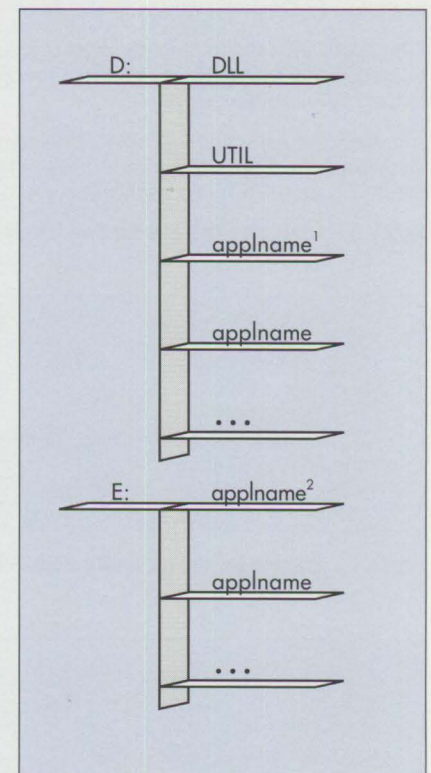
Figure 3. CONFIG.SYS File for a Single Drive

Next, edit the `CONFIG.SYS` file only once, as shown in Figure 3, so that it reflects the suggested directory tree.

If you have only one disk drive (C:) for both OS/2 and applications, put your applications into `C:\APPS`, and substitute `C:\APPS\` where you see `D:\` in the above examples.

#### Running Applications Using Two Drives

Two drives may be necessary in some circumstances, such as a LAN environment. Assume drives D: and E: are used for applications. The suggested directory trees are shown in Figure 4.



Notes:

<sup>1</sup> `D:\appName` stores all personal files, including customized control files, related to the `appName` application.

<sup>2</sup> `E:\appName` stores all original files that came with the `appName` application (.EXE, .DLL, .INF, .HLP, and so on).

Figure 4. Directory Trees for Two Application Drives

```

E:
CD E:\appName
D:
CD D:\appName
SET the variables for the application (if any)
E:\appName\APPLEXE
EXIT

```

**Figure 5. Starting Batch File Where Applications Reside on Two Drives**

```

LIBPATH=.;E.;D:\DLL;C:\OS2\DLL;... 1
SET PATH=E.;D:\UTIL;C:\OS2;... 2
SET DPATH=E.;C:\OS2;... 2
SET HELP=E.;C:\OS2\HELP;
SET BOOKSHELF=E.;C:\OS2\BOOK;

```

**Notes:**

The explanation following Figure 3 (the CONFIG.SYS file for a single drive) also applies to Figure 6. However, in Figure 6:

<sup>1</sup> An application that is running in an individual session will first search for its DLLs in the E:\appName directory before proceeding to search the other DLL directories in the LIBPATH= statement.

<sup>2</sup> In the SET PATH= and SET DPATH= statements, the current directory D:\appName that stores the customized control files will be searched before the directory containing the files from the original application, E:\appName.

**Figure 6. CONFIG.SYS File for Two Drives**

```

C:
CD C:\APPS\appName
D:
CD D\appName
SET the variables for the application (if any)
C:\APPS\appName\APPLEXE
EXIT

```

**Figure 7. Starting Batch File Where Data Files Reside on One Drive**

```

LIBPATH=.;C.;D:\DLL;C:\OS2\DLL;...
SET PATH=C.;D:\UTIL;C:\OS2;...
SET DPATH=C.;C:\OS2;...
SET HELP=C.;C:\OS2\HELP;
SET BOOKSHELF=C.;C:\OS2\BOOK;

```

**Figure 8. CONFIG.SYS File Where Data Files Reside on One Drive**

Figure 5 shows how to create the starting batch file D:\UTIL\appName.COM for each application.

Next, edit the CONFIG.SYS file only once, as shown in Figure 6, so that it reflects the suggested directory trees.

If there is only one additional physical drive (D:), and you want to separate the data files from the application files, put the applications into C:\APPS. But you must change the starting batch files and CONFIG.SYS file. For this situation, Figure 7 shows the starting batch file and Figure 8 shows the CONFIG.SYS file.

*Claude Goffin is an information systems consultant in charge of PS/2 usability projects in the IBM North-West (Europe) organization. He joined IBM Belgium in 1961 in the Scientific Computing Centre. He has been a manager since 1970 in the areas of systems engineering, marketing, installation support, personal computer marketing support, and end-user support. Claude also was a senior staff member in IBM's European Systems Research Institute (IEC), concentrating on end-user computing and programmable workstations. He has an MS in engineering from the Liège University.*

# Creating Resizable Pushbuttons

**Narsu V. Tatikola**  
**Computer Aid, Inc.**  
**Allentown, Pennsylvania**

*Resizable pushbutton windows are often created with a special register class, but this is difficult to do. This article presents an alternate way to create a resizable pushbutton – as a child window of a standard window. The article shows each step in building the subclass window procedure that creates resizable pushbuttons.*

**P**resentation Manager (PM) application programmers often say that it is difficult to create a resizable window with `WC_BUTTON`, a pushbutton register class. But you also can use `WC_BUTTON` to create a resizable pushbutton window. Creating a resizable pushbutton window with `WC_BUTTON` is not as difficult as maintaining the state of the pushbutton when the user interacts with it. This article shows how to create a pushbutton as a child window of a standard window and how to give the window all the functionality it needs – resizing, repainting, and moving the pushbutton along with the parent window.

## Creating a Child Window

The best place to create a child window – in this case, a pushbutton – is within the `WM_CREATE` message in the client window procedure, as shown in Figure 1.

## Varying the Pushbutton's Size and Position

We want to vary the size and position of the pushbutton according to the variations in its parent's size and position. The `WM_SIZE` message within the client window procedure, shown in Figure 2, resizes the pushbutton according to the size of the client.

In the call to `WinSetWindowPos`, the values of `xPos` and `yPos` give the position of the pushbutton relative to the lower left corner of the client. In

the same call, `cXfact` and `cYfact` represent the ratio of the sizes of the pushbutton and the client area. The actual size of the pushbutton can be computed dynamically by multiplying the current client size by the corresponding ratio factors.

## Giving the Pushbutton a Title

We want the title of the pushbutton to be "OK", but for now it has been deliberately set to null in the `WinCreateWindow` call. This is because we can control only the size of the pushbutton and not the size of the text in it. The pushbutton can continue to get smaller, because it is in a window whose size can get smaller. The text does not get correspondingly smaller, so it is possible that the size of the pushbutton will end up smaller than the size of its title.

The simplest way to solve the title problem is to subclass the pushbutton's default window procedure and trap the `WM_PAINT` message. A good place to do the subclassing is within `WM_CREATE` in the client window procedure, immediately after the `WinCreateWindow` call, as shown in Figure 3.

In Figure 3, `ButtonCtrlProc` is a global variable that holds the address of the default window procedure, and `ButtonWndProc` is the subclass window procedure for the pushbutton. (Programming standards often pro-

hibit using global variables. Instead of using a global variable for the address of the default window procedure, use the window-words of the pushbutton to pass the address to the subclass window procedure.)

## Preliminary Subclass Window Procedure

We have developed the first pass of our subclass window procedure `ButtonWndProc`, as shown in Figure 4.

The subclass window procedure in Figure 4 traps the `WM_PAINT` message of the pushbutton and paints the title on the pushbutton using any font of any size. The subclass procedure calls a function `PaintPushButton` that gets the size and the handle of the pushbutton. With this information, it is not difficult even for a novice PM programmer to choose a corresponding font size and type to use for the text. This kind of subclassing works fine for creating a pushbutton with a title, or whenever the painting is done again due to resizing of the parent window.

## Painting the Pushbutton

Painting the pushbutton is a tricky process, especially when the user clicks a mouse on it. When the mouse is clicked on the pushbutton, PM creates an illusion of the button being physically pressed and released. By default, the pushbutton is always in the released state, so the procedure in Figure 4 handles the released state. When the pushbutton is in a pressed state, PM paints the pushbutton in a different style – it creates an illusion of a three-dimensional button being pressed, then adds the title to the pushbutton. To simulate this, we must follow the same process. It is not sufficient to trap `WM_PAINT`; we also have to trap `BM_SETHILITE` – a message that is generated when the pushbutton is suppressed – and then let the process flow in the same

```

case WM_CREATE:
.
.
.
hWndButton = WinCreateWindow (hWnd,          /* Client handle */
                              WC_BUTTON,    /* Register Class */
                              "",           /* Window - Title */
                              BS_PUSHBUTTON,/* Button style */
                              0, 0,        /* Coordinates */
                              0, 0,        /* Size */
                              hWnd,       /* Parent window */
                              HWND_TOP,   /* Position */
                              PBD_OK,     /* Window ID */
                              NULL, NULL);
.
.
.
break;

```

**Figure 1. The WM\_CREATE Message in a Client Window Procedure**

```

case WM_SIZE:
.
.
.
scxClient = SHORT1FROMMP (mp2);
scyClient = SHORT2FROMMP (mp2);
.
.
.
WinSetWindowPos(hWndButton,          /* Pushbutton handle */
                HWND_TOP,           /* Relative position */
                xPos, yPos,         /* Actual position */
                cXfact * scxClient,
                cYfact * scyClient, /* calculated size of pushbutton */
                SWP_SHOW | SWP_SIZE | SWP_MOVE);
.
.
.
break;

```

**Figure 2. The WM\_SIZE Message in a Client Window Procedure**

```

case WM_CREATE:
.
.
.
hWndButton = WinCreateWindow (...);
ButtonCntrlProc = WinSubclassWindow (hWndButton, ButtonWndProc);
.
.
.
break;

```

**Figure 3. Modified WM\_CREATE in a Client Window Procedure**

```

MRESULT EXPENTRY ButtonWndProc(HWND hWnd,
                                USHORT msg,
                                MPARAM mp1,
                                MPARAM mp2)
{
    RECTL rClient;

    switch(msg)
    {
        case WM_PAINT:
            /* Get the size of the pushbutton, resize the font */
            /* according to the size, and paint the text. */
            WinQueryWindowRect( hWnd, &rClient);
            PaintPushButton(hWnd, rClient);
            break;

        default:
            return((*ButtonCntrlProc)(hWnd, msg, mp1, mp2));
    }
    return(FALSE);
}

```

Figure 4. Preliminary Subclass Window Procedure ButtonWndProc

```

MRESULT EXPENTRY ButtonWndProc(HWND hWnd,
                                USHORT msg,
                                MPARAM mp1,
                                MPARAM mp2)
{
    RECTL rClient;
    BOOL bRetVal;

    switch(msg)
    {
        case BM_SETHILITE:
        case WM_PAINT:
            /* Let the default window procedure paint first. */
            bRetVal = (BOOL) SHORT1FROMMP((*ButtonCntrlProc)
                                         (hWnd, msg, mp1, mp2));
            /* Get the size of the pushbutton, resize the font */
            /* according to the size, and paint the text. */
            WinQueryWindowRect(hWnd, &rClient);
            PaintPushButton(hWnd, rClient);
            return(MPFROMSHORT(bRetVal));

        default:
            return((*ButtonCntrlProc)(hWnd, msg, mp1, mp2));
    }
    return(FALSE);
}

```

Figure 5. Final Subclass Window Procedure ButtonWndProc

fashion. We let PM paint the pushbutton first, creating the illusion of a 3-D button being pressed, and then we paint the title on the pushbutton. This involves a small change in the subclass procedure, which is shown in Figure 5 in its final form.

*Narsu V. Tatikola has been a software engineer at Computer Aid, Inc. since 1990. Mr. Tatikola has worked on the design and implementation of a PM-based user interface to a cooperative processing application for a steel manufacturer. Currently, he is developing an LU6.2/APPC-based network application for a cable company. Mr. Tatikola has a Bachelor of Engineering degree in electronics and communication engineering from Osmania University, India and an MS in computer science from Ohio University.*

*Computer Aid, Inc.  
1209 Hausman Road  
Allentown, PA 18104  
(215) 395-5120  
Fax: (215) 366-1566  
E-mail:  
cai-usd!narsu@moravian.edu*

# Configuring Parallel Ports for OS/2

Frank J. Schroeder  
IBM Corporation  
Boca Raton, Florida

*This article explains parallel port configuration for OS/2, including how parallel-port address ranges and hardware interrupt levels have evolved and how some manufacturers have extended the standard. It reviews how DOS and OS/2 drive parallel ports, and how bus architectures affect the use of parallel ports. Finally, the article gives remedies for configuration problems.*

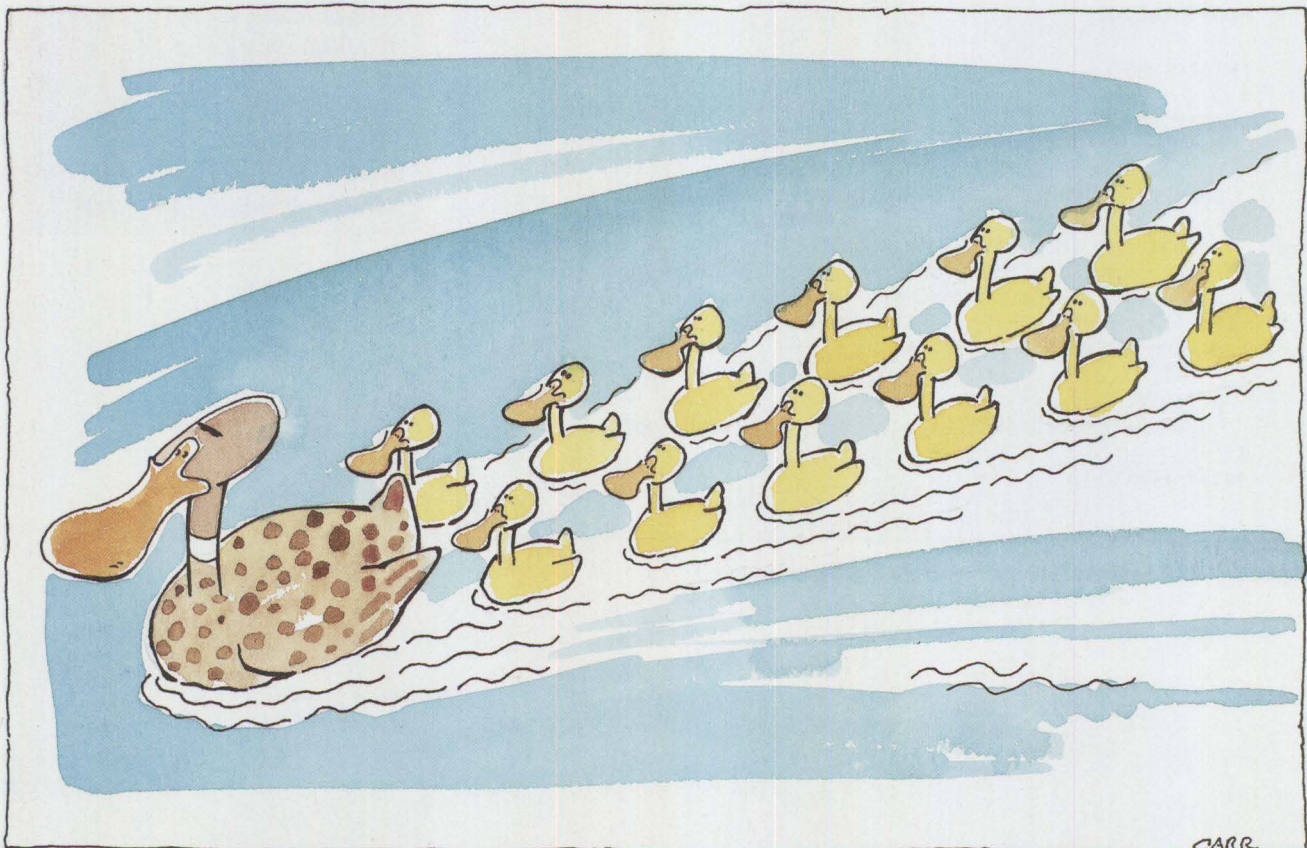
**W**hen upgrading from DOS to OS/2, some users encounter configuration problems that affect printers attached to parallel ports. At first glance, the problem may appear to be with OS/2, because the hardware worked correctly under DOS. In reality, the parallel port may

never have been configured appropriately under DOS, but since DOS never used the incorrectly configured options, the error was not apparent. OS/2 takes advantage of certain parallel port features that DOS does not, so often the solution is simply to configure the parallel ports properly for OS/2.

## Evolution of Parallel Ports

Parallel-port address ranges and hardware interrupt levels have evolved over time. The IBM Personal Computer, introduced in 1981, came with a Monochrome Display/Printer Adapter. This combination adapter was designed for driving a monochrome monitor and for attaching parallel port devices, such as the IBM 5152 Graphics Printer. On this adapter, the parallel port was designed to use the 3BC port address range and the primary parallel port hardware interrupt level, IRQ7. Both the address range and hardware interrupt level were fixed; they could not be altered because there were no DIP switches or jumpers on the adapter card.

As the need arose to attach more serial and parallel devices to system units, IBM introduced the Serial/Parallel Adapter. This adapter allowed users to configure the address ranges and hardware interrupt levels of the



CARR

serial and parallel ports; with this capability, two Serial/Parallel adapters could be installed in one system. The J1 (serial) and J2 (parallel) jumpers on the adapter could be set to two different settings, as shown in Figure 1. The first parallel port setting was for the 378 port address range and the primary parallel-port hardware interrupt level (IRQ7). The second setting was for the 278 port address range and the alternate parallel-port hardware interrupt level (IRQ5).

The combination of one Monochrome Display/Printer Adapter and two Serial/Parallel Adapters enabled the use of three parallel ports in one system and created the *de facto* standard for AT-bus systems running DOS today.

Figure 2 shows the parallel port address ranges and hardware interrupt levels for the three parallel ports on the IBM Personal Computer AT bus.

In 1987, IBM introduced the Personal System/2<sup>®</sup> and Micro Channel architecture. In the PS/2, IBM carried forward the PC AT's port address ranges and hardware interrupt levels to maintain compatibility with existing software. Prior systems had required the purchase of adapter cards to gain features such as a serial or parallel port, an auxiliary port, or a video port. All these ports, standard equipment in PS/2 Micro Channel systems, improve performance and value. The system board parallel port can have any setting described in Figure 3. (Currently, IBM does not produce a parallel port adapter for Micro Channel systems. However, several adapter manufacturers have filled this niche.) Micro Channel adapter cards can be configured to any of the settings in Figure 3 as long as the address range does not conflict with any other installed parallel port.

More recently, IBM has introduced PS/2 systems that contain the AT

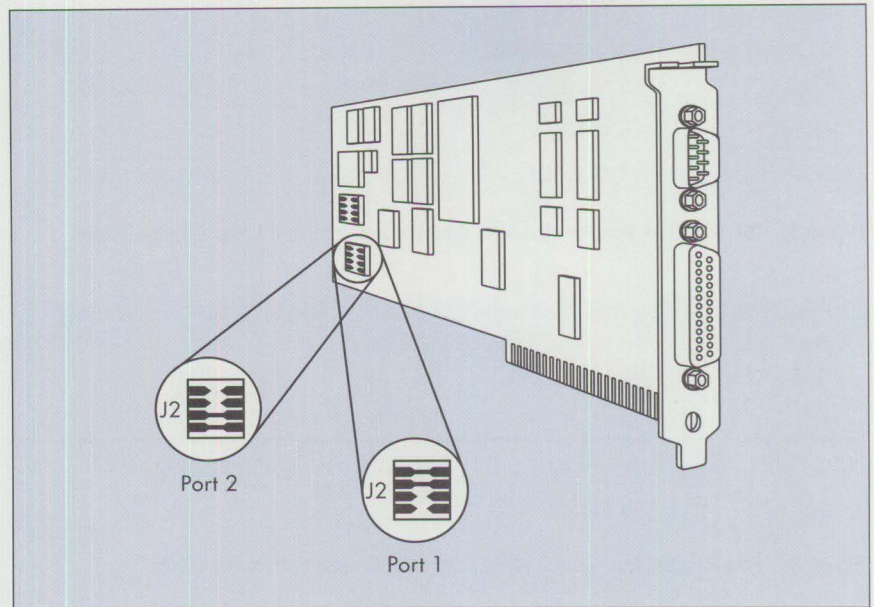


Figure 1. IBM Serial/Parallel Adapter

Logical Name	Adapter Name	Address Range	Interrupt Level
LPT1	PARALLEL1	3BC-3BE	IRQ7
LPT2	PARALLEL2	378-37A	IRQ7
LPT3	PARALLEL3	278-27A	IRQ5

Figure 2. IBM Personal Computer AT Bus Parallel Port Address and Interrupt Level Configurations

Logical Name	Adapter Name	Address Range	Interrupt Level
LPT1	PARALLEL1	3BC-3BE	IRQ7
LPT2	PARALLEL2	378-37A	IRQ7
LPT3	PARALLEL3	278-27A	IRQ7

Figure 3. IBM PS/2 Micro Channel Bus Parallel Port Address and Interrupt Level Configurations

bus. Unlike the original PC AT, these systems include the same built-in ports found in Micro Channel systems, again offering improved performance and value. To configure system board and adapter parallel ports in a PS/2 system that has an AT bus, see Figure 2.

### Extensions by IBM-Compatible Manufacturers

IBM personal computers are open systems. This means IBM publishes

its hardware and software interfaces so that other manufacturers can provide special-purpose adapter cards and software to run on IBM systems. Some of these manufacturers also produce IBM-compatible personal computers that can use these adapter cards and software. Some manufacturers extend the definition of the parallel port for assigning either the IRQ5 or IRQ7 hardware interrupt level to any of the three parallel port address ranges, as shown in Figure 4.

Logical Name	Adapter Name	Address Range	Interrupt Level
LPT1	PARALLEL1	3BC-3BE	IRQ5 or IRQ7
LPT2	PARALLEL2	378-37A	IRQ5 or IRQ7
LPT3	PARALLEL3	278-27A	IRQ5 or IRQ7

Figure 4. ISA and EISA Bus Parallel Port Address and Interrupt Level Configurations

Logical Name	Adapter Name	Address Range	Interrupt Level
LPT1	PARALLEL1	3BC-3BE	IRQ7
LPT2	PARALLEL2	278-27A	IRQ5
LPT1	PARALLEL2	378-37A	IRQ7
LPT2	PARALLEL3	278-27A	IRQ5

Figure 5. Two Parallel Port Combinations for an ISA System Running OS/2

This is done primarily in computers based on Industry Standard Architecture (ISA), which is compatible with the PC AT bus and with those based on Extended Industry Standard Architecture (EISA).

### Assignment of Logical Device Names

No matter which combination of address range and hardware interrupt level is configured, DOS and OS/2 assign the first installed address range in these tables to logical device name LPT1, the next to LPT2, and so on. For example, if the 3BC address range is configured, the operating system assigns it to logical device LPT1. If the 3BC address range is not installed, but the 378 address range is installed, the operating system assigns 378 to logical device LPT1. If the 278 address range is also installed, the operating system assigns it to logical device LPT2 because it is next in the table. Only two parallel ports are configured, so no parallel port corresponds to LPT3. Since data directed to LPT3 cannot be physically transmitted, the operating system discards it.

### DOS Parallel Port Configuration

DOS uses the Basic Input/Output System (BIOS) interrupt 17h function to transmit data through the parallel port to a printer. The BIOS uses the port address ranges to program the parallel port hardware. A technique called *polling* transmits data to the parallel-attached device. The polling technique does not use the hardware interrupt capability of the parallel port, so DOS ignores the interrupt levels. Although some manufacturers extend the hardware interrupt capability of the parallel port, this does not adversely affect DOS users.

### OS/2 Parallel Port Configuration

Unlike DOS, OS/2 takes advantage of the hardware interrupt capability of the parallel port. In OS/2, both the port address and the hardware interrupt level are important. OS/2 runs on AT, EISA, and Micro Channel bus systems. AT-bus (ISA) systems use edge-triggered interrupts, while EISA and Micro Channel systems use level-sensitive interrupts. On ISA systems, hardware interrupt levels cannot be shared by devices, but on EISA and Micro Channel systems the

hardware interrupt levels can be shared. The caveat is that ISA adapters cannot share interrupt levels when installed in EISA bus systems.

### OS/2 and ISA Systems

To install one parallel port on an ISA system running OS/2, configure the parallel port with any address range and interrupt level pair, shown in Figure 2.

To install two parallel ports on an ISA system running OS/2, refer to the valid combinations for two parallel ports, as shown in Figure 5. (Because ISA systems do not support hardware interrupt sharing, the two port configurations shown in Figure 2 that use interrupt level IRQ7 should not be used together.)

To install three parallel ports on an ISA system running OS/2, configure the adapters exactly as specified in Figure 2. As you may have noticed, Figure 2 shows two devices using hardware interrupt IRQ7. This seems contradictory, because ISA systems cannot share hardware interrupt levels. How can both work?

The OS/2 parallel-port device driver is responsible for the algorithm used to transmit data through the parallel port. When a print request is issued to LPT1 and LPT2 is idle, there is no conflict with the hardware interrupt level. The request to print on LPT1 can proceed using IRQ7. Conversely, if a print request is issued to LPT2 and LPT1 is idle, there is still no conflict with the hardware interrupt level. The request to print on LPT2 can proceed using IRQ7. But what happens when LPT1 is printing and, before it completes, a request to print is issued to LPT2 (or vice versa)?

To run the second device (LPT2) that is requesting the hardware interrupt level IRQ7 while the first one (LPT1) is still running, the OS/2 parallel-port



device driver initially uses the OS/2 timer services. The OS/2 timer services use the real-time clock and generates hardware interrupts at a rate of 18.2 times per second. This interrupt frequency is significantly slower than the interrupt frequency achievable using a device-generated hardware interrupt level. If the LPT2 request is left to run only on the timer services, the request would take an unacceptably long time to complete. To work around this resource conflict, when the interrupt-based request (LPT1) completes transmission of its buffer, the device driver switches the timer-based request (LPT2) to the hardware interrupt level. When the first request (LPT1) returns with more data to transmit, the device driver assigns it to the system timer. It stays there until the second request (LPT2) – which is now at the hardware interrupt level – completes. This process of switching print requests between the hardware interrupt level and the system timer continues until the resource conflict ceases.

If the request assigned to the hardware interrupt level stops transmitting because of device errors (such as out of paper, offline, and power off), then that request is similarly switched to the system timer. This frees the (faster) hardware interrupt level for use by a request from another device. That request will then operate at maximum throughput.

### **OS/2 and EISA Systems**

One, two, or three EISA parallel ports can be configured to run under OS/2. How these parallel ports are configured depends on which other adapters are running on the same hardware interrupt level, their interrupt sharing capability, and the interrupt sharing capability of the parallel port adapter. If the parallel port or other adapter cards on this level cannot share interrupts, then the parallel port must be configured as specified in

Figure 2. Hardware interrupt level contention must be resolved. If the parallel port and other adapters on the same hardware interrupt level can share interrupts, then the parallel port can be configured to use either interrupt level, as specified in Figure 4. Be sure to make the correct decisions when using EISA configuration software. Since that software cannot detect the presence of ISA hardware, it does not prevent a configuration mistake.

### *Micro Channel adapter configuration is easier than EISA adapter configuration.*

EISA configuration software lets you set any of the possible configuration alternatives. However, it will not provide an alert when you have made an error. If the hardware does not support the alternative selected, OS/2 may not be able to operate the adapter card. For example, if a parallel port adapter does not support hardware interrupt sharing, only one adapter can be assigned to that particular hardware interrupt level. If a parallel port adapter uses address range 278 and supports interrupts on IRQ5 only, but the user sets the interrupt level to IRQ7 through the set configuration software, then OS/2 will incorrectly drive the adapter from IRQ7. It is important to know whether the adapter supports hardware interrupt sharing, on what level, and what address range it uses.

### **OS/2 and Micro Channel Systems**

To install one, two, or three parallel ports on a Micro Channel system and run them with the OS/2 parallel-port device driver, the adapters must be

configured as specified in Figure 3. Any combination of the address ranges (except duplicates) can be specified if all the hardware interrupt levels are specified as IRQ7.

The Micro Channel bus architecture supports hardware interrupt sharing. The hardware interface used by the device driver to transmit the data on Micro Channel parallel ports can specify whether the device is generating an interrupt. This hardware feature enables interrupt sharing to work. The OS/2 parallel port device driver takes advantage of this feature. When a print request is issued to any of these devices, the OS/2 parallel port device driver uses IRQ7 to transmit the data.

There is an exception to the IRQ7-only restriction. If a Micro Channel adapter manufacturer provides both a parallel port adapter that can interrupt on IRQ5 and a loadable OS/2 parallel-port device driver, then the adapter can be configured to run on IRQ5 under OS/2. Follow the manufacturer's recommendations for loading the loadable OS/2 parallel-port device driver. Most Micro Channel adapters that can interrupt on IRQ5 can also interrupt on IRQ7; consequently, they can be run by the OS/2 parallel-port device driver.

Micro Channel adapter configuration is easier than EISA adapter configuration, because there is no need for concern about which adapters support interrupt sharing – they all do with Micro Channel.

### **Configuring the Parallel Port**

There are several suggested actions to correct a parallel port problem. First and easiest is to check the configuration of the parallel ports installed in your system. If you are running on an ISA system, turn off the power, remove the cover of your system unit, and check the DIP

switches and jumpers. The documentation that accompanies the parallel-port adapter card or the system unit should show the appropriate settings for DIP switches and jumpers.

EISA and Micro Channel systems have a software configuration option. In Micro Channel systems, software can customize the configuration, so there is no need to power off the system unit or remove the system unit cover to check or modify the configuration. Just insert the reference (configuration) diskette that comes with the system unit into the A: drive and reboot. You will see a menu containing an option to set the configuration. Follow the directions given with the Set Configuration program to configure the parallel ports appropriately. EISA systems can be similarly configured by software unless an ISA adapter is installed; ISA adapters must be configured by physically modifying DIP switches or jumpers.

If a parallel port adapter is installed after the system unit, use the EISA configuration file (.CFG) or the Micro Channel Adapter Description File (.ADF) distributed with the new adapter to configure it. There is an option to copy the particular file from the adapter's diskette to the system reference diskette. During system configuration, when the Set Configuration program cannot find the file on the reference diskette, it prompts you to insert the diskette that contains the file. If you have two parallel ports installed, make sure the adapters do not use the same address range. The PS/2 Set Configuration program provides alerts about conflicts like this, whereas EISA configuration software cannot.

### Direct Memory Access

The system-board parallel port on PS/2 Models 56, 57, 80-A21, 80-A31,

90, and 95 supports a feature called Direct Memory Access (DMA). DMA provides increased throughput and system performance. Running on Micro Channel systems, OS/2 recognizes and automatically takes advantage of DMA parallel support if the DMA feature is configured properly. (DOS does not exploit the DMA parallel feature.) The Set Configuration option, found on the main menu of the PS/2 Reference Diskette, can verify that the DMA option is supported and enabled. The Set Configuration option displays the current system setup, and in particular, the parallel port name and parallel-port arbitration level (found under Built-In Features). By default, the parallel port is set to PARALLEL1 (3BC and IRQ7) and its arbitration level is set to SHARED7, which means DMA is enabled. Cycle through the available arbitration level options and select the DISABLE option to disable the parallel port DMA. Using the DMA parallel option is highly recommended on PS/2 systems that support it.

### OEM Incompatibilities

Two other scenarios, categorized as Original Equipment Manufacturer (OEM) incompatibilities, are worth noting. The first problem occurs when the OEM parallel port does not generate hardware interrupts, generates them incessantly, or generates them spuriously. This results when the parallel port hardware does not contain interrupt logic or the logic is faulty. The second problem occurs when the parallel port generates hardware interrupts on a nonstandard hardware interrupt level and the manufacturer does not provide a way to modify the level (DIP switches, jumpers, or a set configuration program). For example, when the manufacturer sets the port address to 3BC or 378 and the hardware interrupt level to IRQ5 on AT or Micro Chan-

nel bus systems, the configuration cannot be changed to resolve the incompatibility. Incompatibilities can occur on either the system board or adapter-card parallel port. Clearly, if the problem is on an adapter card, it is a less expensive and a less disruptive problem to fix.

Alternatives exist when a parallel port has interrupt problems. The system board or adapter-card parallel port can continue to be driven in a non-interrupt fashion by DOS. Another alternative is to contact the manufacturer to explain the problem, determine whether the manufacturer is aware of the problem, and ask if a revision is available.

### Conclusion

Parallel port configuration under OS/2 may be an issue of concern. Fortunately, configuration is generally a simple process. Usually it takes just a few minutes to change DIP switches or jumpers, or to run a program to set the configuration. In other cases the hardware may not be truly IBM-compatible, and will not run under OS/2 without specialized (nonstandard) device drivers to support the parallel ports.

*Frank J. Schroeder is a staff programmer in the OS/2 Device Drivers and Multiple Virtual DOS Machines (MVDM) department. Frank has worked on OS/2 since Version 1.0 and is currently working on performance and functional enhancements to the OS/2 Printer Subsystem. He has a Bachelor of Technology degree in computer science from Rochester Institute of Technology and an MBA from the University of Miami.*

# Performance Characteristics of ES 1.0 Database Manager

Benetta N. Perry, Harry Shelin Chen, Bob Russell, and Timothy J. Li  
IBM Corporation  
Austin, Texas

*This article describes some performance characteristics of the IBM Extended Services (ES) 1.0 Database Manager running with OS/2 2.0. It gives insights into system performance through various test cases, benchmarks, and "what-if" scenarios. The information in this article should help users configure and use ES 1.0 more efficiently.*

This article describes some performance characteristics of Database Manager in IBM Extended Services for OS/2 1.0 (ES 1.0) running with OS/2 2.0:

- **Single workstation scenario:** Performance measurements are made on key database functions for local and remote operations. Sensitivity studies show the effects of tuning some parameters.
- **Transaction processing applications:** Multi-user transaction processing measurements are made based on the book order benchmark defined by the National Software Testing Laboratories (NSTL). The effects of parameter tuning and communication protocol on product performance are demonstrated. Comparisons with the predecessor, IBM OS/2 Extended Edition 1.3, also are made.
- **Forward recovery:** This is a new set of functions in ES 1.0 Database Manager. The key questions about the performance of forward recovery are "How much does system performance degrade because of archival logging?" and "How long does it take to perform forward recovery when needed?" These questions are addressed using one of NSTL's benchmarks.

## System Configuration

Measurements presented in this article use a 33 MHz IBM PS/2 Model 95 for the database server. The amount of memory depends on the test cases: for the single workstation scenario, it is 16 MB; for transaction processing applications it is 32 MB. These memory sizes are not required for the test cases; they were chosen to prevent memory overcommitment during the benchmarks.

The database server and its clients are connected by a 4 MB token ring.

All benchmarks are performed using the ES 1.0 Database Manager product running with the OS/2 2.0 operating system.

## Performance in Single Workstation Scenarios

The single client scenario uses a 16 MHz PS/2 Model 80-071 with 8 MB of memory and one 115 MB disk. NetBIOS is the communication protocol.

Most of the database and Database Manager configuration parameters are set with their default values. The exceptions are that the maximum storage of the lock list is changed from the default of 8 to 25 4-KB pages, and the database heap size (DBHEAP) is changed from the default of 3 to 10 segments. During the performance measurements, some configuration parameters are varied to demonstrate their effects on performance.

All the database tables use the same table definitions and contain the same number of fields, as shown in Figure 1. The differences in the tables are in the number of rows in a table, the indices defined, and the range of data values for the fields. The table sizes

Column	Width	Type	Ordering
Field1		Integer	Ordered
Field2		Integer	Unordered
Field3		Integer	Unordered Rep
Field4	21	Character	Ordered
Field5	21	Character	Unordered
Field6	21	Character	Unordered Rep
Field7	5,0	Decimal	N/A
Field8		Scientific	N/A
Field9		Date	Random
Field10		Integer	Ordered
Field11		Integer	Unordered
Field12		Integer	Unordered Rep

Figure 1. Database Table Definitions

Function	Local	Remote
Update 1,000 rows	10.5 seconds	10.8 seconds
Insert 1,000 rows	12.1 seconds	13.2 seconds
Delete 1,000 rows		
– without index	4.8 seconds	5.1 seconds
– with index	32.8 seconds	32.8 seconds

Figure 2. Basic Functions that Change the Database

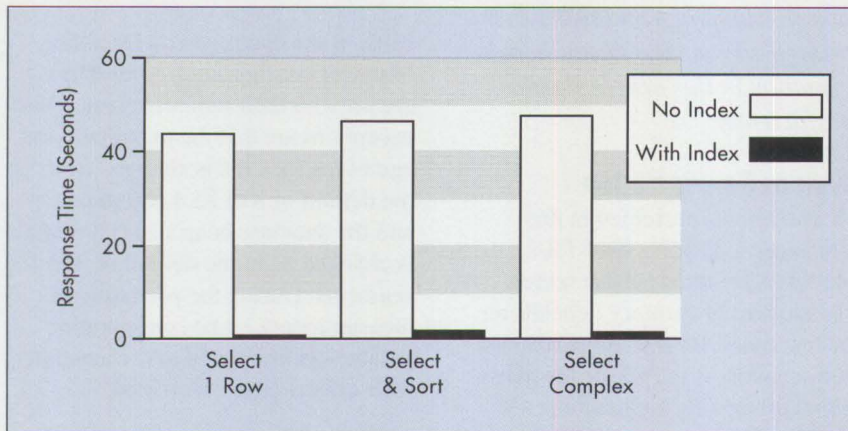


Figure 3. Effects of Indexing on Selects

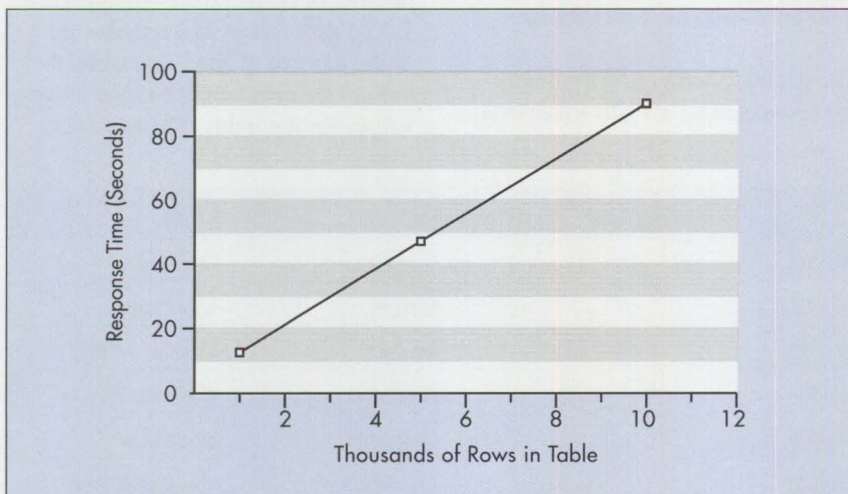


Figure 4. Effects of Table Size on Remote Select All

used in the measurement include 10,000, 20,000, 30,000, 40,000, 50,000, and 100,000 rows. Indices are defined on the first three fields of the tables.

#### Implementation of Test Cases

The test cases are implemented in the C language with embedded Structured Query Language (SQL) statements. Within each test run, a Start Using Database command is issued before each query begins, and a Stop

Using Database command is issued after each query ends. This causes the buffer pool to be emptied, ensuring the consistency of the results. The time recorded is the elapsed time of the query, excluding the time for the Start and Stop Using Database commands.

#### Base Measurement Results

Figure 2 shows performance results for the basic functions of changing the database with Update, Insert, and Delete. The Update is performed on an indexed 10,000-row table. The Insert is from an indexed 10,000-row table into an empty table. In all cases, the differences in response time are small between local and remote operations because these operations pass only a small amount of data between the client and the server.

Deleting 1,000 rows with an index takes substantially more time than without an index. When deleting with an index, changing indices by restructuring the B-tree takes a large amount of time, although the index helps to quickly find the rows to be deleted.

Figure 3 shows performance results of various forms of selects from a 100,000-row table. The Select One Row case has a simple predicate. The Select Complex case has a complex predicate that returns 20 rows. The Select and Sort case evaluates a complex predicate and then sorts the selected rows; it also returns 20 rows. Note the drastic difference in performance between the indexed and non-indexed cases. These results clearly demonstrate the importance of proper index design.

Generally, the elapsed time increases as the amount of data returned increases. Figure 4 shows the effect of table size for a Remote Select of all rows in the table. The time increases linearly as the size of the table increases. For this test, record blocking is used, with a block size of 4 KB.

Figure 5 shows the elapsed time for a select with an Order By query that returns the entire table. Rows are stored in the same order as that specified in the Order By clause, so the sort time is minimized. Still, selects on indexed tables are consistently faster than on tables without indices. If an index (on the column given in the Order By clause) exists, then rows can be retrieved in the desired order with no sorting at all.

Figure 6 shows the elapsed time of a two-table join. As the result of joining the table that has 20,000 rows with the table that has 30,000 rows, about 30,000 rows are selected. Response time is plotted against buffer pool size for tables with and without indices.

With the default buffer pool size (1,000 KB), the join with index takes longer than without index. This is caused by the extra Input/Output (I/O) for reloading the index into the buffer pool when the buffer pool size is inadequate. However, when the buffer pool size increases, the elapsed time for the indexed case starts to improve, while the performance for the non-indexed case remains almost constant. When the buffer pool is large enough, the indexed case outperforms the non-indexed case, as expected.

### Utilities

Several utilities are described below.

**Create Index:** Figure 7 shows the performance of Create Index for two cases – when the records are sorted on the index key, and when the records are in random order. For sorted input, the elapsed time is less. It is also proportional to the size of the table. Sorted inputs minimize the I/O needed to build the B-tree for the index. When the field is unsorted, much I/O may be needed to get the nodes to insert when building the B-tree. The performance of Create

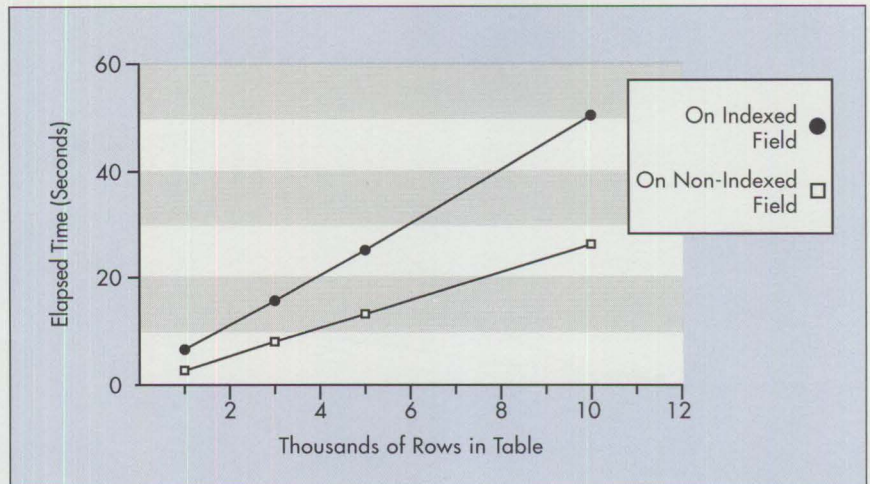


Figure 5. Select All Rows with Order By Query

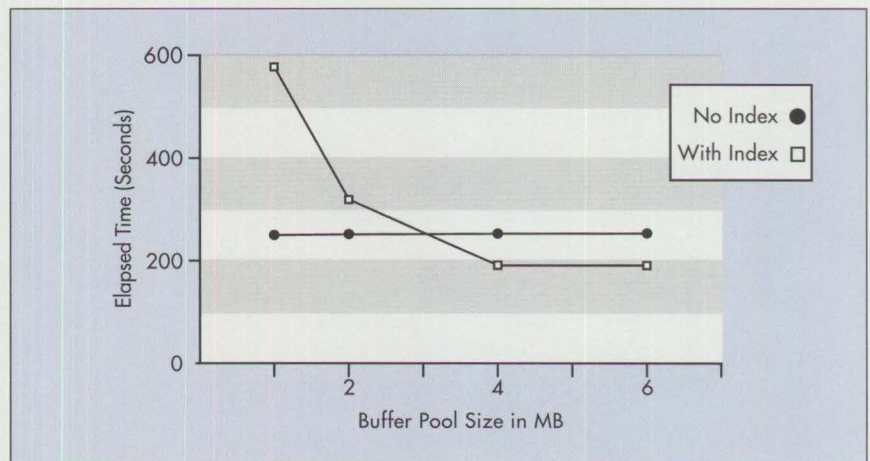


Figure 6. Effects of Indexing and Buffer Pool on a Two-Table Join

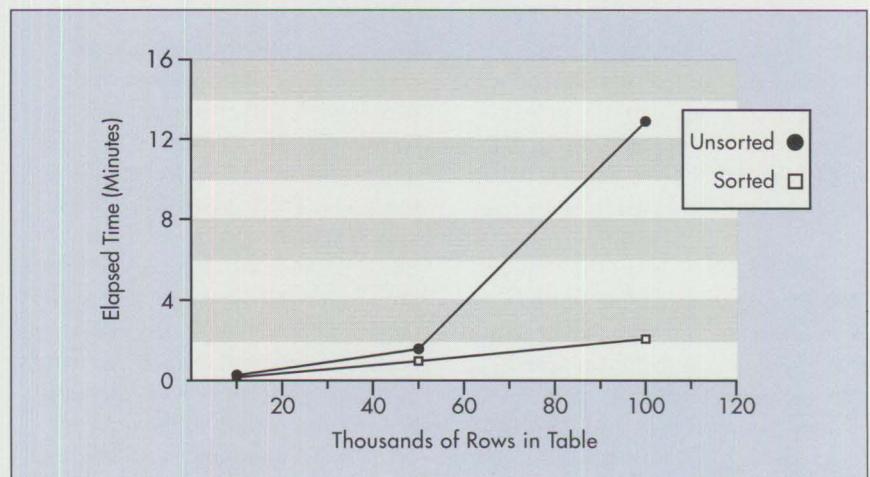


Figure 7. Create Index for Sorted and Unsorted Records

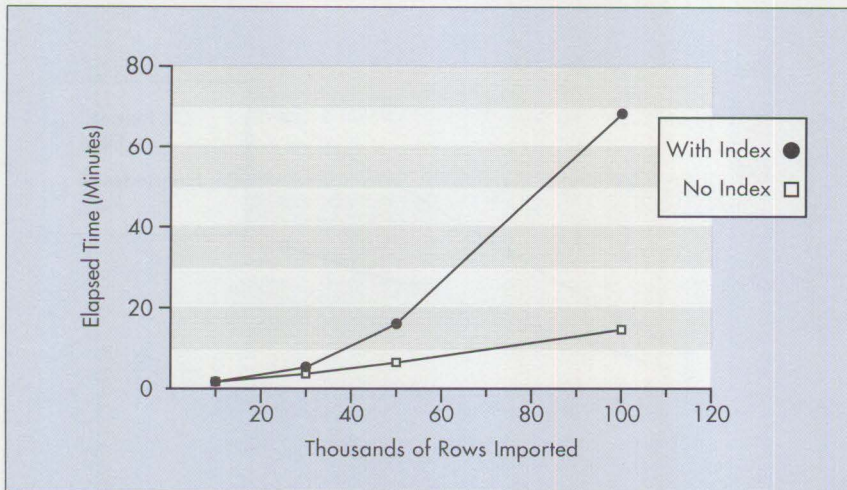


Figure 8. Import Without Index and With Index

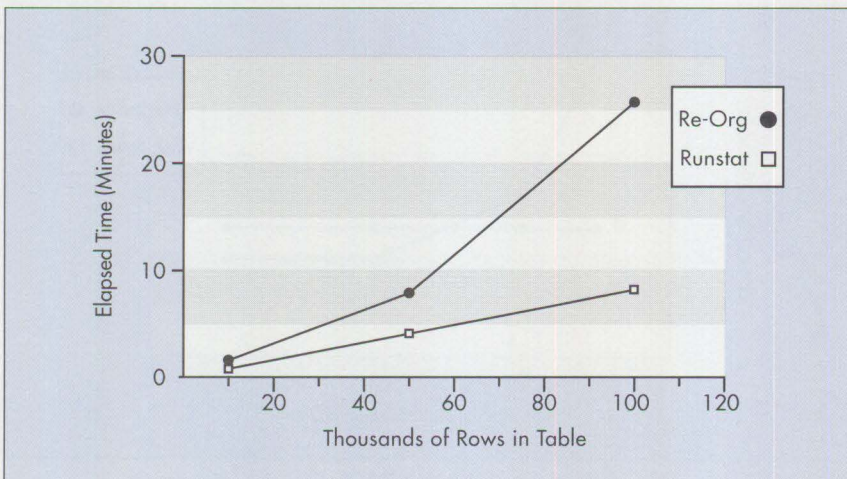


Figure 9. Reorg and Runstats Utilities

Index depends on many factors, including the number of rows in the table, the complexity of the index, the length of the indexed fields, the size of the buffer pool, and the randomness of the indexed fields.

**Import:** Figure 8 shows the elapsed time for importing a table from a delimited ASCII file into the database. If no index is defined, the elapsed time increases linearly as the number of imported rows increases. With an index, the elapsed time is longer than the non-indexed case, and it increases faster than linearly. Additional processing is required to build the B-tree

for the index. In the non-indexed case, the new table can be written to disk sequentially, whereas each new record added to an indexed table requires additional disk I/O to update the index. The increase in response time with table size is greater than linear because the depth of the B-tree grows with the table size, increasing the average number of disk operations needed to add each record. The extent of the increase depends on the number and complexity of the indices defined, the randomness of the data in the indexed fields, and the size of the buffer pool.

**Reorg and Runstats:** Figure 9 shows the performance of the Reorg and Runstats utilities. The time for Runstats increases linearly as the number of rows in the table increases, while Reorg increases more drastically. This is expected; because Runstats goes through the table to gather statistics uniformly, the elapsed time should be proportional to the size of the table and its indices. For Reorg, however, the system rebuilds the table for data and the B-tree for indices. The rebuilding of the B-tree generally causes a more drastic increase as the table size increases.

### Parameter Tuning

In ES 1.0 Database Manager, many database configuration parameters can be easily modified with the configuration tool. Thus, users can select parameter values that are well suited to their applications. The following examples show how tuning some configuration parameters improves performance.

**Buffer Pool:** Figure 10 shows the transient behavior for repeated single random selects. The operation starts with an empty buffer pool. Average response time is calculated for every 100 random selects. The average response time for each 100 selects gradually decreases, then stabilizes. Initially the buffer pool is empty, so the chance of finding the randomly selected row is low. Then, as more data and index pages are read into the buffer pool, the hit ratio increases and average response time drops. The best steady state results are attained when all the data and index pages within the selected range are in the buffer pool – that is, with an almost 100% hit ratio.

### Effects of Buffer Pool on Random Selects

Figure 11 shows the effects of a buffer pool on random selects with various ranges. A single select is made from a 100,000-row table with index. Each curve in this figure

represents random selects from a different range of field values. The field for the select criterion follows the natural ordering of the data in the table. Thus, the range for the random select determines the portion of the table that is accessed.

For selects in the range 1 to 10,000, a 2,000 KB buffer pool is more than sufficient to hold all the data and indices. The hit ratio is already 100%. Further increasing the buffer pool size will not improve the performance. For selects in the range 1 to 100,000, the maximum buffer pool size of 6,000 KB still cannot hold all the data and index pages. Thus, the hit ratio is far less than 100%.

The buffer pool size should be estimated by running benchmarks that reflect the real database applications.

**Record Blocking:** Record blocking often improves performance of remote selects from a client by reducing the number of messages sent between the client and the database server. Figure 12 shows the elapsed time of selecting all rows of a table as a function of the block size for the data transfer. A block size of zero means no record blocking is used. The most dramatic improvement is from no record blocking to using record blocking with a block size of 4 KB. Further increases in block size have only marginal return in performance. The elapsed time is proportional to the amount of data transferred.

### Performance for Transaction Processing Applications

The hardware configuration is equivalent to the setup that NSTL used for their recent measurement report ("SQL Servers for LANs," *Software Digest Rating Report*, 1991, Volume 8, Number 15.). The database server is a 33 MHz IBM PS/2 Model 95 with 32 MB of memory.

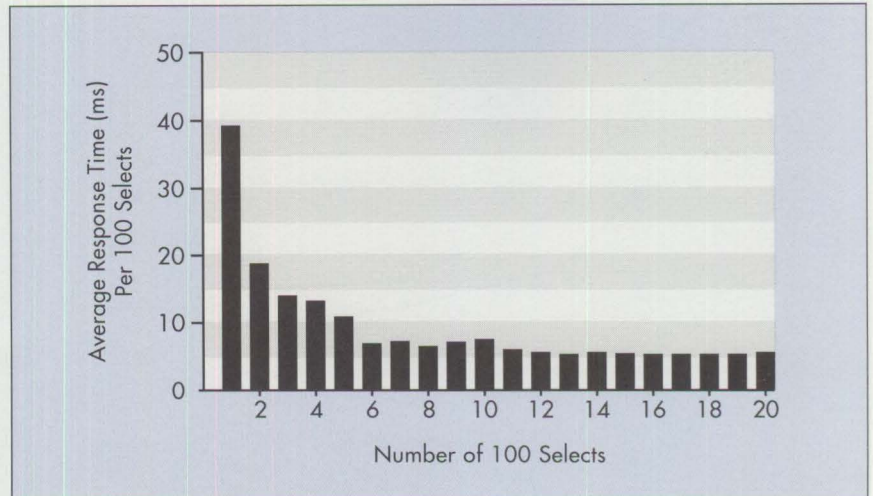


Figure 10. Transient Behavior of Random Select

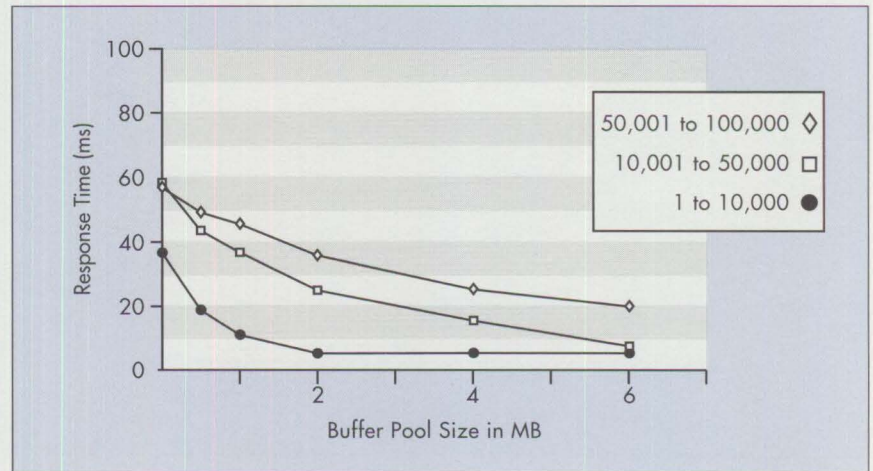


Figure 11. Select on Random Key Value with Various Ranges

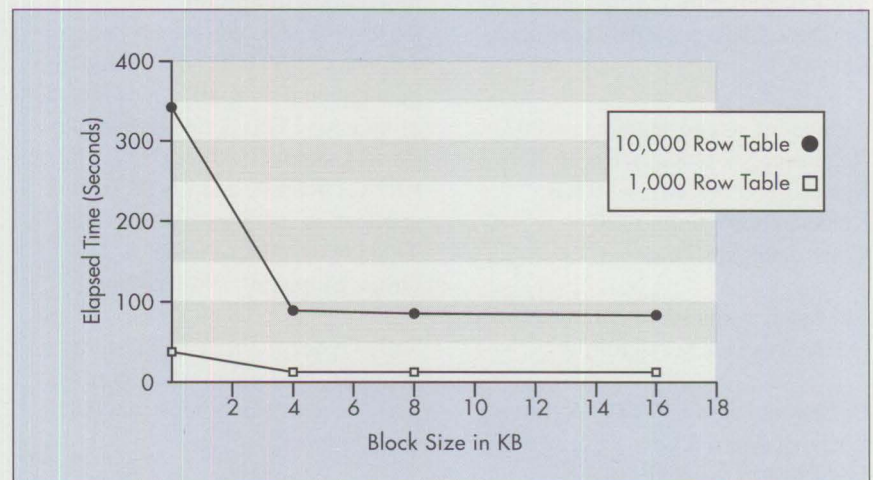


Figure 12. Effects of Record Blocking

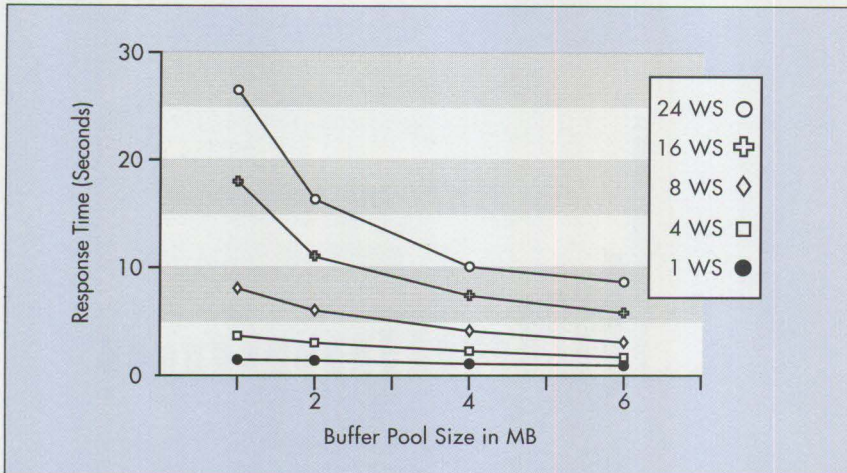


Figure 13. Effect of Buffer Pool Size on ISBN Order

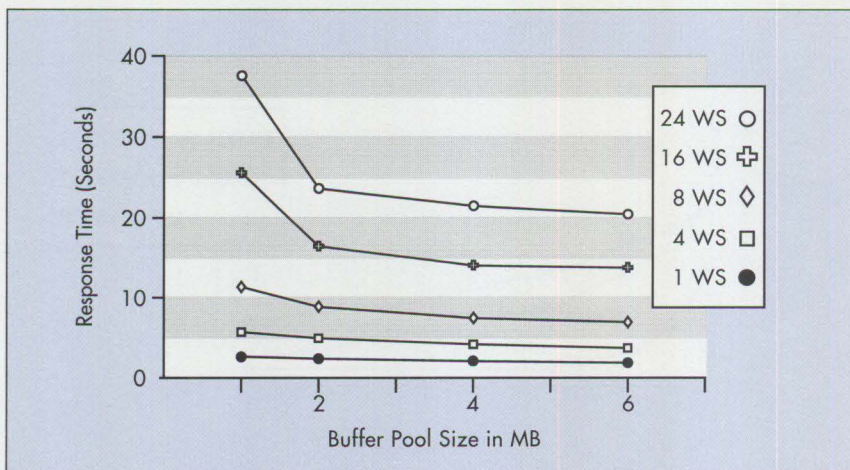


Figure 14. Effect of Buffer Pool Size on Author Order

The key configuration parameters for Database Manager (DBM) are set as follows.

Number of shared segments = 802  
 Requester I/O block size = 16 KB  
 Server I/O block size = 16 KB  
 Remote connections = 120  
 Communication heap = 3

The key configuration parameters for the database are set as follows:

Buffer pool size = 1,500 4-KB pages  
 Lock list size = 25  
 Log file size = 1,000  
 Number of primary logs = 4  
 Number of secondary logs = 0

#### Benchmark Definition

To test the SQL database server performance, NSTL has developed the database and its applications for book order entry. This database includes the key tables BOOK, AUTHOR, LINK, ORDER, and ENTRY. The application includes five scenarios: Title Order, Author Order, ISBN Order, Shipment, and Payment.

**Title Order** scenario is a test that simulates customers who know only part of the title of a book that they want to order.

**Author Order** scenario is a test that simulates customers who know only part of the author's name.

**ISBN Order** scenario is a test that uses an International Standard Book Number (ISBN) to order the book desired.

**Shipment** is a program that finds the lowest order number for an unshipped order. It selects the ORDER record for that number and all corresponding entries, and updates the ORDER record. The program also updates BOOK records with ISBN numbers that correspond to the selected entry records.

**Payment** scenario selects the ORDER record and its corresponding entries. The ORDER record is updated.

Where applicable, an Application Remote Interface (ARI) is implemented for better performance. For each scenario, multi-user cases are tested. No "think time" is inserted between the transactions. For more details about this benchmark, see the NSTL measurement report cited above.

#### Base Measurements

Figures 13, 14, and 15 show the effects of buffer pool size on the response time for the ISBN Order, Author Order, and Payment scenarios, respectively. For the first two scenarios with many users, the performance is very sensitive to the buffer pool size. However, in the Payment scenario, performance is much less sensitive to the buffer pool size. This demonstrates again that the effect of a buffer pool depends on the nature of the scenario being run.

Figure 16 shows the effects of communication protocols on the response time for the Author Order scenario. It indicates that for this scenario, Advanced Peer-to-Peer Network (APPN) has a slight edge over NetBIOS. Generally, when a small amount of data is transferred, APPN often performs better than NetBIOS. The default protocol is NetBIOS, which is easy to configure and to get the system started. When the difference in per-



formance becomes significant, it may be desirable to spend time and effort to configure for the APPN protocol.

Figures 17 through 21 compare the performance of ES 1.0 with its predecessor, OS/2 Extended Edition (EE) 1.3. Clearly, the current system consistently outperforms its predecessor. The improvement is very significant with a large number of users. The main reason for this improvement is the difference in how the base operating system addresses memory. The predecessor system can address only 16 MB of memory, even though more physical memory is installed in the server hardware. With the buffer pool set at 6 MB and a large number of users, OS/2 EE 1.3 overcommits memory, extensive swapping occurs, and performance degrades drastically. Because OS/2 2.0 can address much larger memory spaces, all the memory available in the hardware can be used. Thus, the response time curve remains linear, even for many users.

#### Measurements with Archive Logging

Forward recovery with archival logging is a new function supported by the ES 1.0 Database Manager. The performance degradation from archival logging is a valid concern. To investigate this effect, the NSTL Payment scenario with sixteen users was run with no "think" time. For this set of tests, there were four primary log files of 200 4-KB pages each.

To establish a base line for comparison, this scenario was run using a circular log. Then the same scenario was repeated with log retain and archival logging with different media. Log retain provides logging on the hard disk instead of circular logging; the hard disk log can be manually archived later (offline) to a different medium. For archival logging, the media used was a 340 MB disk in the server, an IBM PS/2 Rewritable Optical Disk (128 MB), and an IBM PS/2 2.3 GB Small Computer Systems

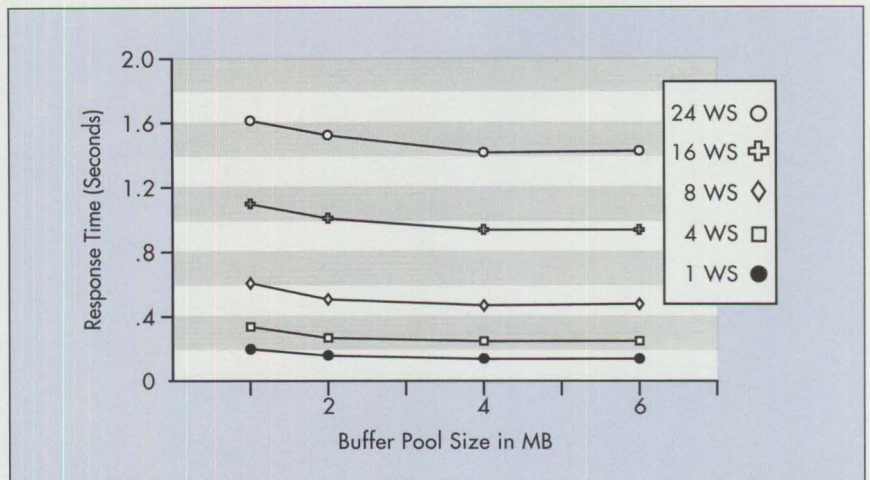


Figure 15. Effect of Buffer Pool Size on Payment

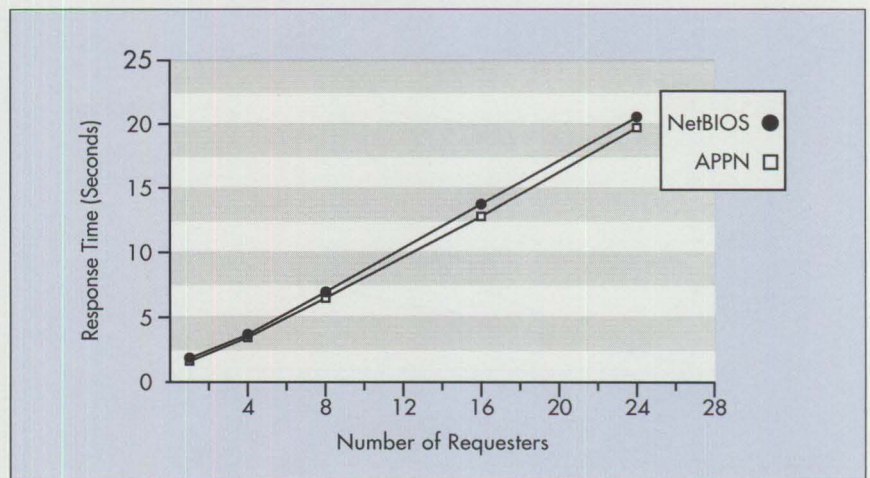


Figure 16. Effect of Communication Protocol on Author Order

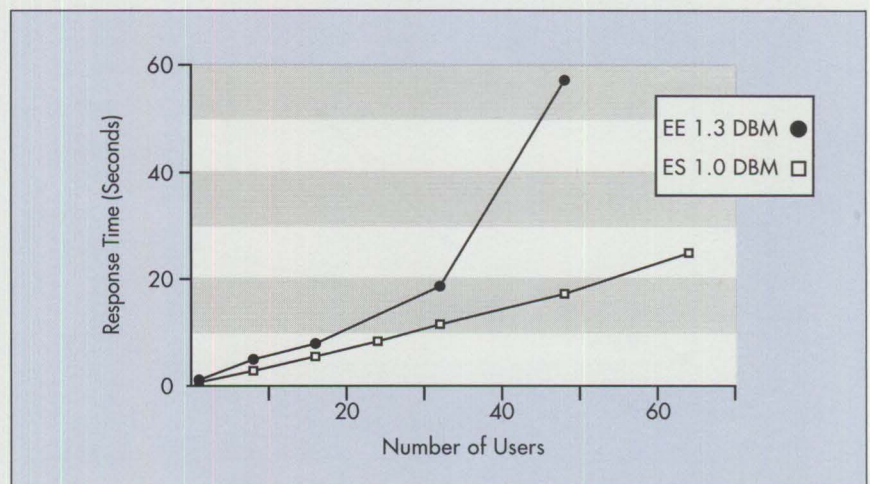


Figure 17. Comparison of ES 1.0 and OS/2 EE 1.3 for ISBN Order

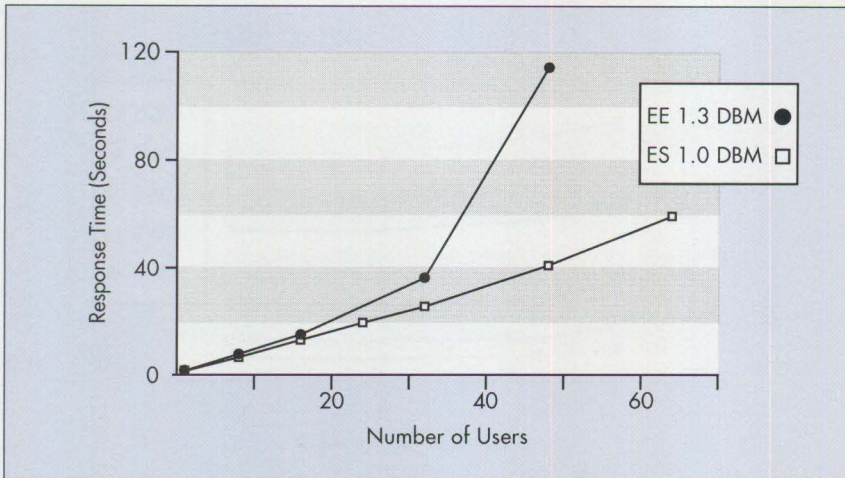


Figure 18. Comparison of ES 1.0 and OS/2 EE 1.3 for Author Order

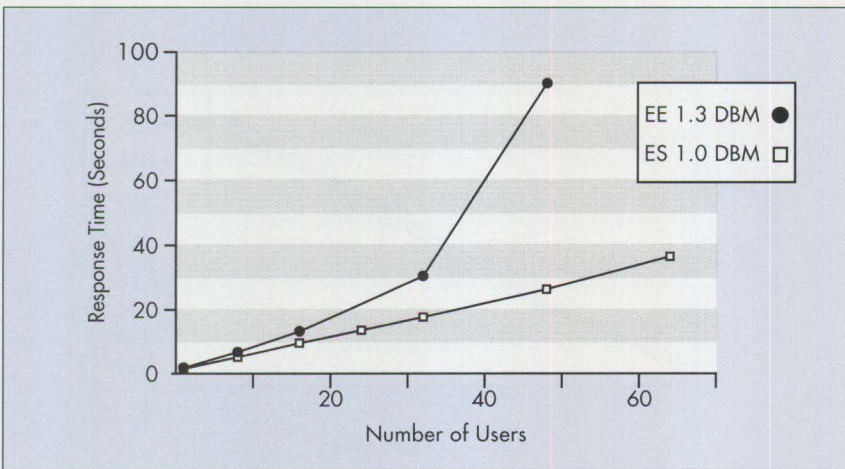


Figure 19. Comparison of ES 1.0 and OS/2 EE 1.3 for Title Order

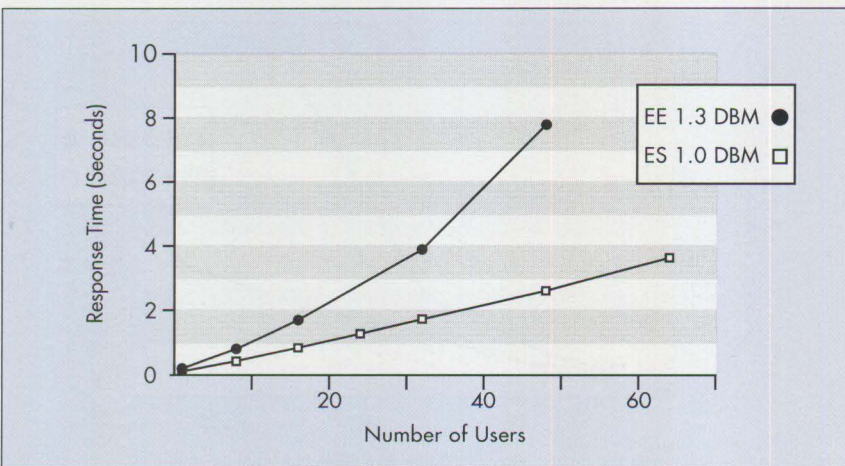


Figure 20. Comparison of ES 1.0 and OS/2 EE 1.3 for Payment

Interface (SCSI) Tape Drive with Sytos Plus® File Backup Manager. For each run, the response time for each user was measured.

Figure 22 shows the effects of archival logging on response time. Note that using a disk as the medium causes the least amount of degradation, and the tape drive causes the most degradation. In all cases, the percent of degradation is fairly low.

To evaluate the performance of forward recovery, NSTL's Payment scenario with sixteen users was run for ten minutes. This produced about five log extents of 800 KB each. To perform forward recovery, the database was restored to the state before the run, then the archived log was "played back" so that it could be rolled forward. Total time was measured using various media. Figure 23 shows the total elapsed time for forward recovery. Recovery from disk takes the least amount of time, the Rewritable Optical Disk is a close second, and the tape drive takes much longer to complete this task. In general, the elapsed time of forward recovery depends on the size of the original database, the activities during the period of archival logging, and the medium used.

## Summary

The benchmarks and test cases described in this article demonstrate the following key points:

- The database design (such as the index structure) can dramatically affect the ES 1.0 Database Manager performance.
- Configuration parameters (such as buffer pool size) or communication protocol can affect performance.
- If OS/2 2.0 is used, the Database Manager can take advantage of a system with more than 16 MB of memory, thus improving performance.

- With NSTL's Book Order benchmark, ES 1.0 with OS/2 2.0 consistently outperforms OS/2 EE 1.3.
- The performance degradation due to archival logging seems reasonably low.
- The elapsed time for forward recovery depends on the medium used.

**Benetta N. Perry** is a senior associate programmer in the IBM Personal Systems Programming Center in Austin, Texas. Her work in LAN Systems Performance focuses on performance analysis of Distributed Systems Services. She has a BS in computer science from Grambling State University in Louisiana.

**Harry Shelin Chen** is a staff programmer in LAN Systems Performance in the IBM Personal Systems Programming Center in Austin, Texas. He is currently working on performance analysis of Distributed Systems Services. He has an MS in computer science from the University of Kentucky.

**Bob Russell** is an advisory programmer working on performance analysis for Database Manager and Distributed System Services in the IBM Personal Systems Programming Center in Austin, Texas. Before taking his current assignment, he worked on analyzing the performance and systems assurance of Database Manager.

**Timothy J. Li** is a senior programmer in the IBM Personal Systems Programming Center in Austin, Texas. Dr. Li is currently a performance analyst on Distributed Systems Services. He has a BS in electrical engineering from National Taiwan University and a PhD in electrical engineering from Purdue University.

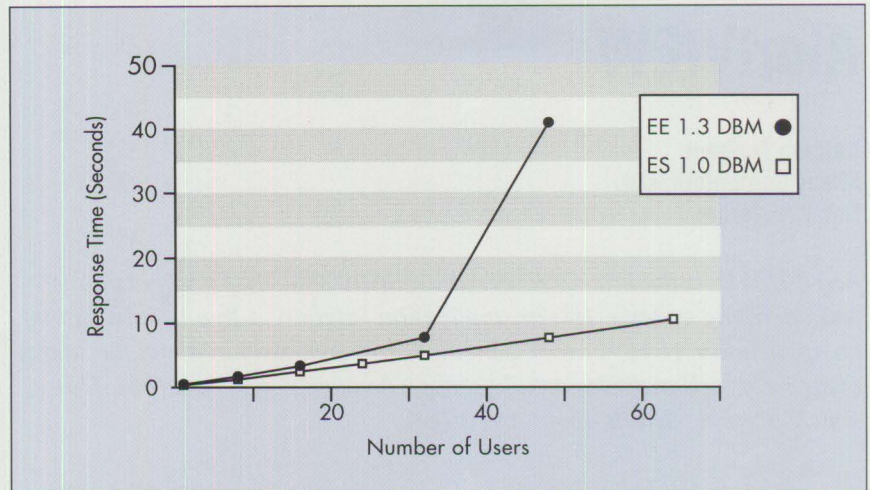


Figure 21. Comparison of ES 1.0 and OS/2 EE 1.3 for Shipment

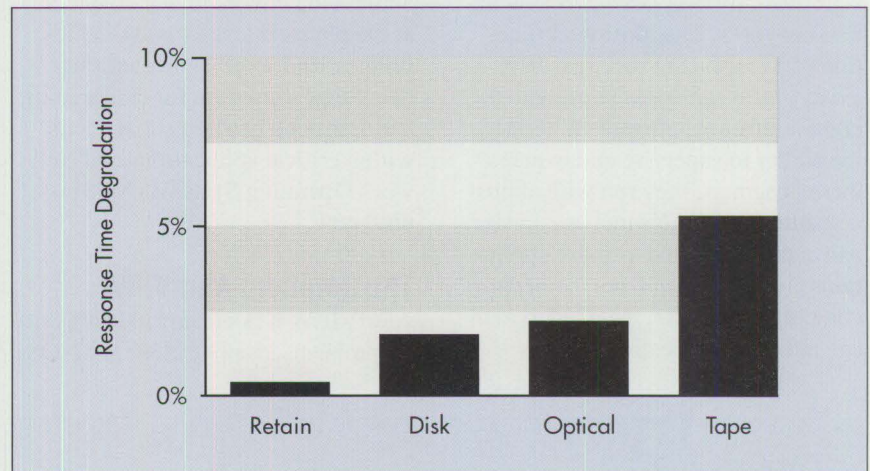


Figure 22. Effects of Archival Logging on Payment

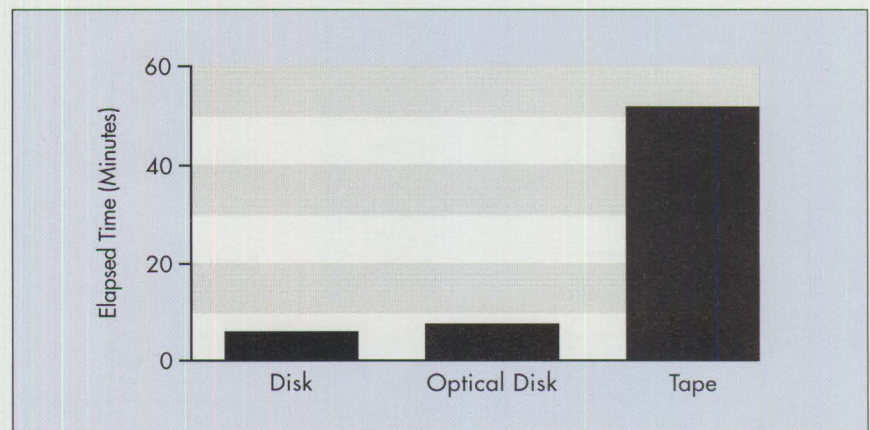


Figure 23. Forward Recovery on Payment

# AlertVIEW

**Shlomo Touboul**  
**Shany Computers, Ltd.**  
**Natanya, Israel**

*AlertVIEW™ is the first network management solution for applications and operating systems. It is an application “sniffer” – that is, it looks into an application, gives network administrators information about the source of an error, and provides details about when and why it occurred. This article presents details about AlertVIEW.*

Local Area Networks (LANs), particularly token-ring networks, are playing increasingly significant roles in large communication networks. The most vital function of a LAN – the very reason it exists – is to run applications and communication software. Without the ability to supervise and oversee these programs, they run with almost no control. As LANs increase in size and complexity, and as more applications are downsized from mainframe computers to LANs, the need to manage network applications from a

central location becomes more pronounced. However, vendors of network management products have addressed network management only at the physical level – hubs, cables, bridges, and so on. No vendor has developed a solution for determining and resolving problems that occur within critical applications and Network Operating Systems (NOSs) – until now.

## The Solution: AlertVIEW

AlertVIEW was created to bring critical problems in applications and NOSs

to the attention of the NetView® administrator at the enterprise level and the IBM Network LAN Manager (NLM). The NLM helps the LAN administrator manage the physical layer of the token ring at the LAN level. AlertVIEW adheres to IBM's Systems Network Architecture (SNA); therefore NLM and NetView can receive alerts using IBM's Network Management Vector Transport (NMVT), a protocol used in SNA to transport information. AlertVIEW also can send alerts to Novell's Network Management System (NMS) using the Simple Network Management Protocol (SNMP).

The seven-layer Open System Interface (OSI) model is a guideline that vendors use to ensure interoperability in the design of hardware and software. Within OSI, AlertVIEW provides software alert management for the logical link level up to the application level. This completes a missing piece of network management.

AlertVIEW detects application, operating system, and communication software problems when they occur. It permits heterogeneous networks, including bridges and gateways, to be viewed simultaneously at a single administrator's station. This alleviates the need to have multiple workstations for managing the network.

## AlertVIEW Overview

AlertVIEW gives network administrators a complete, unrestricted view of all facets of network application activities and failures. It provides a workstation agent that monitors application and operating system activities. This agent module, called AlertVIEW Station, detects application failures when they occur – before users are aware of problems. The AlertVIEW alert reports details about the workstation, application name, operating system resources (such as system version, memory, file name, directory

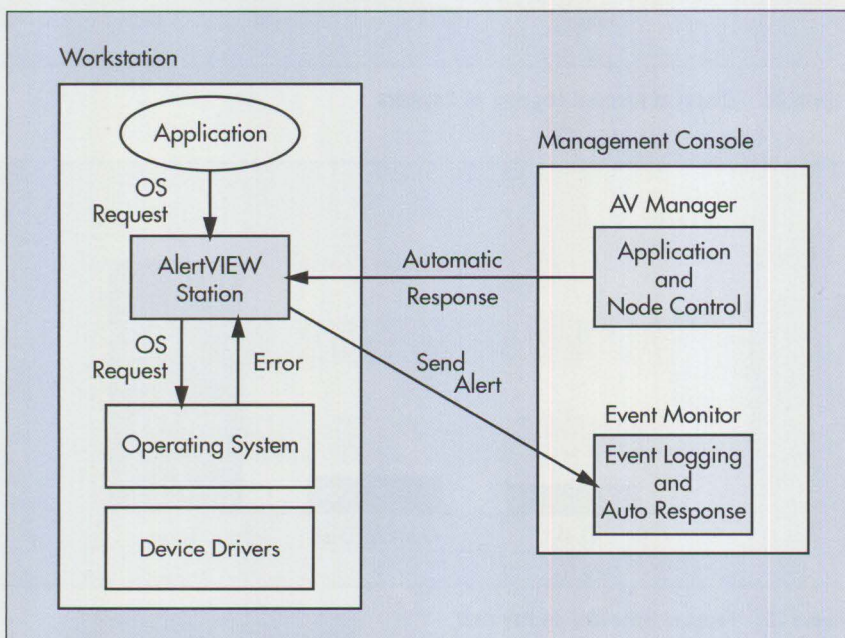


Figure 1. AlertVIEW Agent and Console

name, disk sector number, and disk head and track), and network operating system resources (such as file server name, queue name, and Internet<sup>®</sup> Packet Exchange/Sequenced Packet Exchange (IPX/SPX) socket number).

Network administrators can configure AlertVIEW to provide an automatic response to predefined problems. This "trigger" ensures minimal damage and prevents problems from spreading. For example, an automatic trigger can occur when an electronic mail gateway fails to access a file

and aborts. AlertVIEW can remotely restart the operation when the problem is corrected.

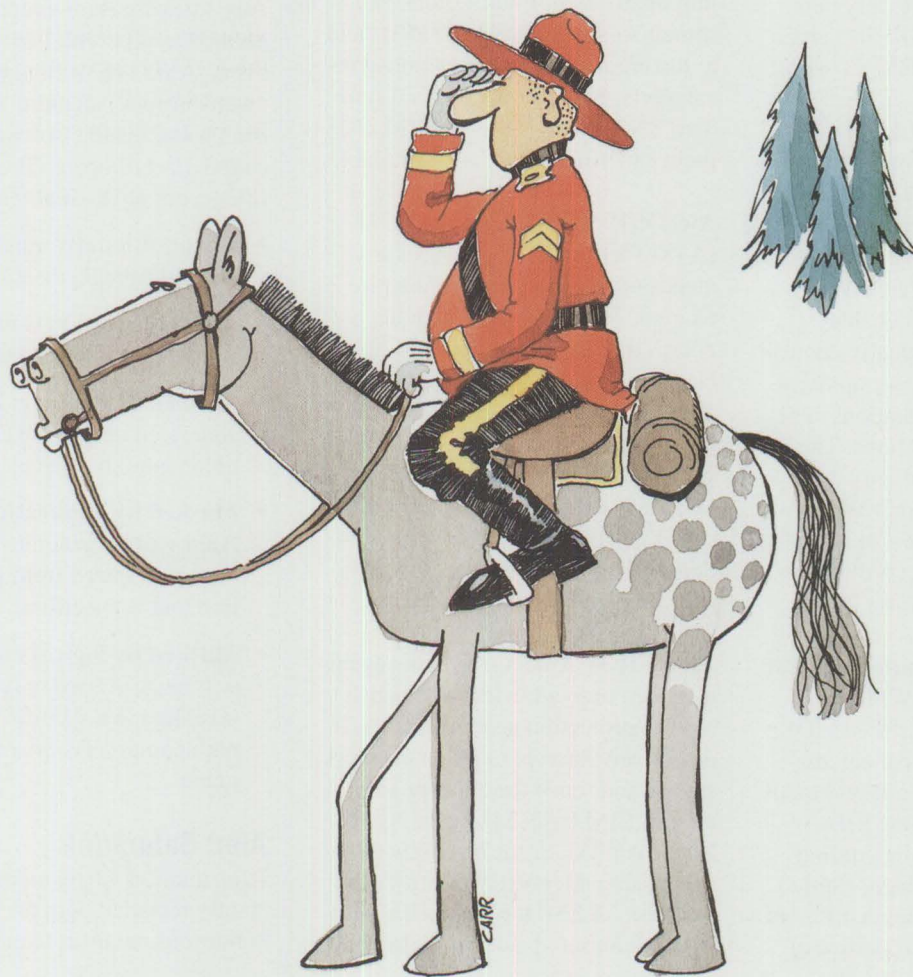
### AlertVIEW Components

AlertVIEW consists of several software components as shown in Figure 1.

**AlertVIEW Station (AVS):** AVS is the agent loaded onto every DOS, Windows, or OS/2 workstation on the LAN. It detects software errors occurring on the workstation, and sends the alerts to IBM LAN Manager, NLM, NMS, and the AlertVIEW Event Monitor (AVEM). Further-

more, if the LAN Manager station is configured to be host-connected, then the NetView operator can receive critical alerts from the AlertVIEW stations. AVS is implemented as a DOS Terminate-and-Stay-Resident (TSR) program, a Windows Dynamic Link Library (DLL), or an OS/2 device driver and process.

The AVS can be configured to include the AlertVIEW Application Programming Interface (API). All problems caused by the operating system's failure to provide service to the application (such as an infinite




loop within the program, or bad logical record contents when reading a file) are automatically reported by AVS to the management console. For more specific control, programmers can use the API to send logical alerts from the application and to monitor them the same way other software alerts are detected by AlertVIEW. This enables all applications on a station to send application-defined NMVT alerts to the IBM LAN Manager or AVEM.

**AlertVIEW OS/2 Server:** This is the agent for OS/2 servers. It is implemented as a LAN service that detects when errors occur in applications running on the OS/2 server, when critical errors and traps occur in OS/2, and when available disk space reaches a predetermined level. The OS/2 Server agent also receives any network alerts from the OS/2 file server and converts them into NMVT alerts.

**AlertVIEW Manager (AVM):** This system management component remotely controls all AlertVIEW agents (AlertVIEW Station, AlertVIEW Anti-Virus, AlertVIEW OS/2 Server, and all future AlertVIEW agents). This component enables administrators to dynamically control all types of event filters, set up users, freeze or unfreeze workstations, and control workstation functions. The AVM software is loaded only onto the network administrator's Windows or WIN-OS/2 workstation so that the AVS filters are set up according to users' specific applications.

**AlertVIEW Event Monitor:** AVEM is an OS/2 or enhanced Windows Multiple Document Interface (MDI) application designed to collect, display, and automatically respond to all AlertVIEW and other IBM SNA alerts. An administrator can define Structured Query Language- (SQL-) like views, each displaying a different set of alerts (for example, a view of

all alerts from a specific application, workstation, or segment). The administrator also can predefine an automatic response – a trigger – for each specific alert. The trigger can modify the workstation's CONFIG.SYS file, forward specific alerts into cc:Mail, or forward alerts from a remote site.



### *AlertVIEW detects application, operating system, and communication software problems when they occur.*

Customization facilities enable setting up different fonts, colors, and sounds to any alert type or alert field to distinguish critical from non-critical alerts. Because AVEM is a standard SNA Event Monitor, IBM code point customization is also available.

**AlertVIEW SNA Gateway (AVGTY):** AVGTY provides a means of transportation for alerts received by the AVEM from the AlertVIEW agents to the IBM host. NetView can monitor any application alert from the LAN, just as it does for mainframe applications. AVGTY is an OS/2 application that requires Digital Communications Associates (DCA<sup>®</sup>)/Microsoft Communication Server, IBM Communications Manager, or a compatible product.

**AlertVIEW Anti-Virus:** This agent is loaded onto all workstations that require protection against viruses. It detects any attempt to infect the workstation and sends the appropriate alert to IBM LAN Manager, NLM, NMS, AVEM, or the NetView console. Using AlertVIEW Anti-Virus with the AVEM, the network administrator can set up automatic triggers

that freeze, reboot, or disconnect a workstation that reports a virus infection. This action automatically prevents the virus from spreading to other workstations and servers.

Figure 2 shows how AlertVIEW modules interrelate.

### **AlertVIEW Filtering**

AlertVIEW features extensive filtering and auto-blocking ("saturation") management. Auto-blocking automatically prevents an alert from being sent over the network, because the software error has been determined to be insignificant. However, saturation values can be set so that the administrator is notified if this problem occurs a certain number of times during a specific period.

Any software error situation is defined as an *event*. For a specific event, AlertVIEW can generate or disable (block) an alert. The ability to enable and disable software alerts is called *alert filtering*. The following filter types are available:

- **Unconditionally masked:** Alert is unconditionally disabled.
- **Unmasked:** Alert is unconditionally enabled (it will be generated).
- **Masked by counter:** Alert will be generated only if the counter threshold (saturation level) is exceeded.
- **Masked by application name:** Alert will be generated only if the alert is detected from an application that is specified.
- **Masked by logical name:** Alert will be generated only if the alert is related to a specific file name, path name, server name, or queue name.

### **Alert Saturation**

In a situation where an event is constantly repeated with no termination conditions, an infinite number of

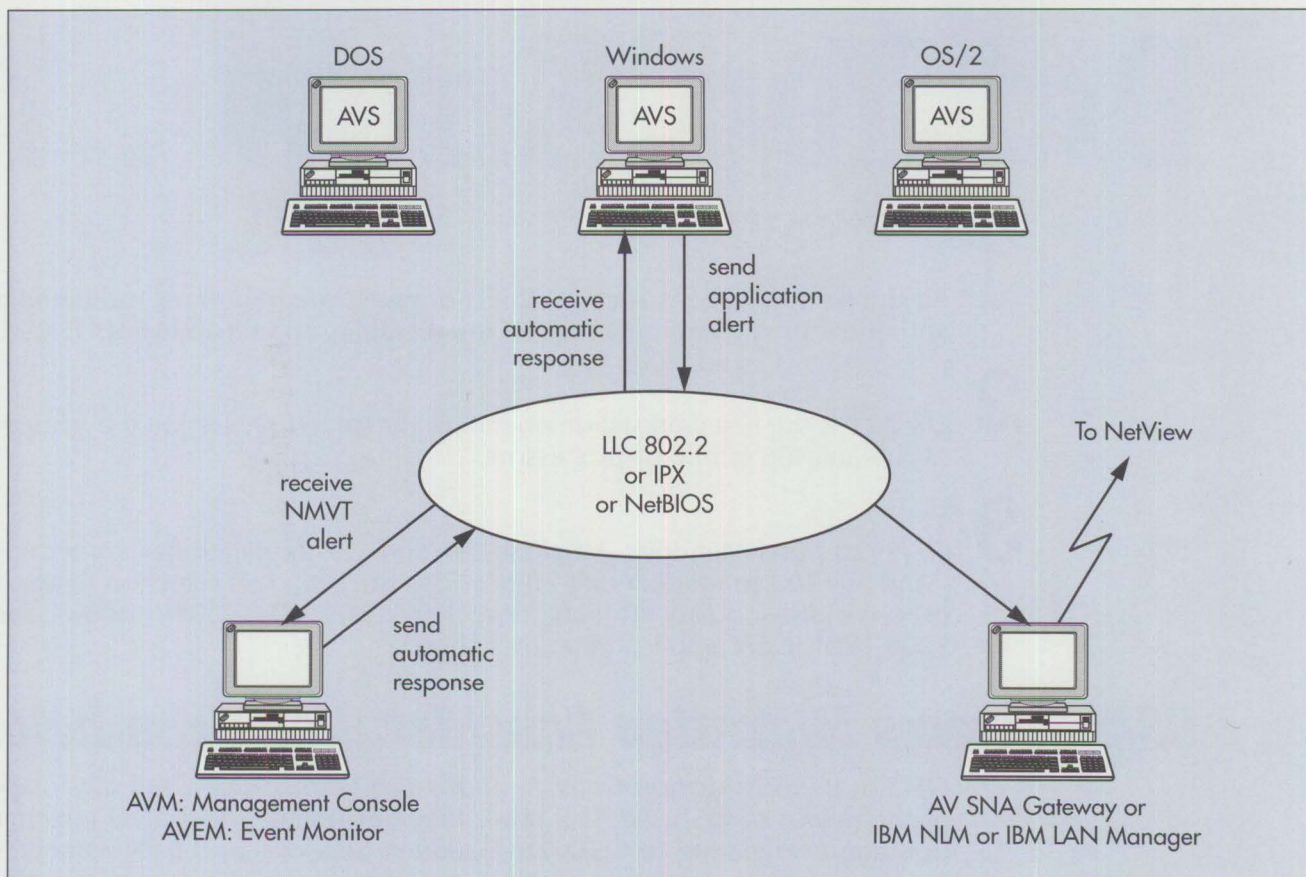


Figure 2. AlertVIEW Modules

software alerts will be sent to the administrator's console, distracting the administrator's attention from other critical alerts. The transmission of an alert ceases automatically if a predetermined counter (determined by either a time value, or a type of alert, or both) is not exceeded.

The AlertVIEW filtering and saturation mechanism reduces the overhead of the network by reducing the number of alerts, and permits concentration on particular categories of problems.

### AlertVIEW Development

AlertVIEW was created and developed by Shany Computers Ltd, a software developer that specializes in support software for local area networks. AlertVIEW was developed in cooperation with IBM, ensuring that every aspect of the program is fully compatible with NetView, IBM LAN Manager, and Network LAN Manager. AlertVIEW is fully compatible with Institute of Electrical and Electronics Engineers (IEEE®) 802.2, and has been designed to work with IBM LAN Server, Novell NetWare, Microsoft LAN Manager, and Banyan® VINES®.

*Shlomo Touboul, president of Shany and founder of the company in 1985, received his degree in computer science from Technion Polytech University in Haifa, Israel.*

*Shany Computers Ltd.  
4 Smilansky Street  
Natanya, Israel 42432  
Phone (972)-53-626201  
Fax (972)-53-342418*

*Shany, Inc.  
2680 Bayshore Parkway, Suite 104  
Mountain View, CA 94043  
(415) 694-7410  
Fax (415) 694-4728*

# Did You Know...?

- 1** Advanced applications such as object-oriented programming, multimedia, and distributed computing require the versatility and reliability of OS/2 2.0.
- 2** OS/2 2.0 is the first workstation operating system to fully exploit the features of the 386/486 family of processors.
- 3** OS/2 2.0 runs Windows Versions 2.0 and 3.0 applications—something Windows 3.0 cannot do. OS/2 2.0 provides the broadest selection of applications in the industry, running more than 20,000 DOS, 5,000 Windows, and 2,500 16-bit OS/2 applications unchanged.

## ***OS/2 Systems Migration Considerations can help!***

*OS/2 Systems Migration Considerations* can help you take full advantage of the power of OS/2 2.0! The book provides detailed technical information about migrating to OS/2 2.0, Extended Services, and LAN Server 2.0 environments. It also covers features of previous OS/2 versions and helps you locate them in 2.0, and describes features of Microsoft Windows included in 2.0.



**Yes...**

Send me \_\_\_\_\_ copies of *OS/2 Systems Migration Considerations* at \$18.95\* plus \$3.95 shipping and handling, a total of \$22.90\*\* per copy.

\*Discounts available for multiple copies.

\*\*California residents add applicable sales tax.

Charge to my VISA/MasterCard       Check       Money Order

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

City/State/Zip: \_\_\_\_\_

Phone: \_\_\_\_\_

VISA/MasterCard #: \_\_\_\_\_ Expires: \_\_\_\_\_/\_\_\_\_\_/\_\_\_\_\_

**Make checks payable to: The TDA Group, P.O. Box 1360, Los Altos, CA 94023. For faster service, order toll-free by calling (800) 551-2832, or fax this form to (415) 948-4280. All orders must be prepaid.**

OS/2 is a registered trademark of the International Business Machines Corporation. Microsoft and Windows are registered trademarks of the Microsoft Corporation.



# Screen Reader/2

**Ron Hoekzema and Kathleen Wahl Bode**  
**IBM Corporation**  
**Boca Raton, Florida**

*Screen Reader/2™ is IBM's latest product for enabling computer use among users who are blind or who have visual disabilities. This article gives an overview of Screen Reader/2 and discusses several ways in which users – including those who do not have visual impairments – can make productive use of Screen Reader/2. The article also discusses the rest of the IBM Independence Series™ of products that assist people who have various kinds of impairments.*

**O**f all the computer tools that convert screen information to speech, Screen Reader/2 is the first tool that makes IBM's Graphical User Interface (GUI) available to users who are blind or have visual disabilities. Screen Reader/2 enables people with visual impairments to enjoy the power and productivity afforded by IBM's GUI and the multi-tasking capability of OS/2 2.0.

In 1988, IBM's Special Needs Systems group announced its first DOS-based product, Screen Reader 1.0. This product converted computer screen text to voice through the use of voice synthesis. Since then, the technology and applications have progressed to GUIs, such as IBM's OS/2.

## Verbalizing the OS/2 2.0 GUI

Screen Reader/2 runs under OS/2 2.0. It provides voice output of all OS/2 screen information, including error and status messages. Screen Reader/2 recognizes and verbalizes the many icons and objects in OS/2 2.0. It functions in all sessions of OS/2, including DOS and Windows sessions.

Screen Reader/2 provides many things that sighted users take for granted. For example, when a user selects "File" from an action bar, a menu is presented. Each item in the menu has one underscored letter, such as O in Open or H in Help. If the user presses the "O" key, the Open command is executed. Screen Reader/2 tells the user what the underscored letters are, thereby assisting users with visual disabilities. By remembering which letters are underscored, the user can increase productivity by bypassing the spoken explanation of what is in the menu.

In Screen Reader/2, the user can create a fast path from task to task. For example, assume that under OS/2 2.0 the user is running an emulator, a spreadsheet application, and a word processor application. Each one can be given an identifier (such as 1 or 2) that can be keyed in on the special 18-key keypad. When the user keys the identifier, the selected application becomes the active screen.

## Following the Visual Focus

This product follows the visual focus – what a sighted person's eyes are

actually following on the display screen – and reads the text or the controls (such as pushbuttons) automatically. It alerts the user when changes occur – for example, when the status message "Mail Waiting" appears in the host emulation session. While browsing through a document, the user can read the entire screen, selected lines, or selected words. Words can be spelled out audibly, and keystrokes can be heard as they are typed.

Users should have little difficulty installing and using Screen Reader/2. Installation instructions are provided on audio cassette, in print, and in Braille. A complete Screen Reader/2 tutorial also is provided online. It is similar to the powerful OS/2 online documentation facility. Although knowledge of OS/2 and related software applications is helpful, a person familiar with using computers under DOS should easily be able to use Screen Reader/2 in the OS/2 environment.

## Other Uses for Screen Reader/2

There are an estimated 11 million people with visual disabilities in the United States. Screen Reader/2 was developed primarily to open doors for them and others worldwide. However, Screen Reader/2 also can be used by sighted people in environments where they can "read" the screen in conjunction with performing another task. For example, in a training environment, a user may look at an assembly or component and perform an operation while listening to instructions spoken by Screen Reader/2. Screen Reader/2 also can be an aid to people who have reading impairments.

The Screen Reader/2 product includes the following components:

- Software
- A separate 18-key keypad for entering user commands
- 3.5-inch and 1.44 MB diskettes that include the online documentation
- Installation instructions (audio cassette, print, and Braille)

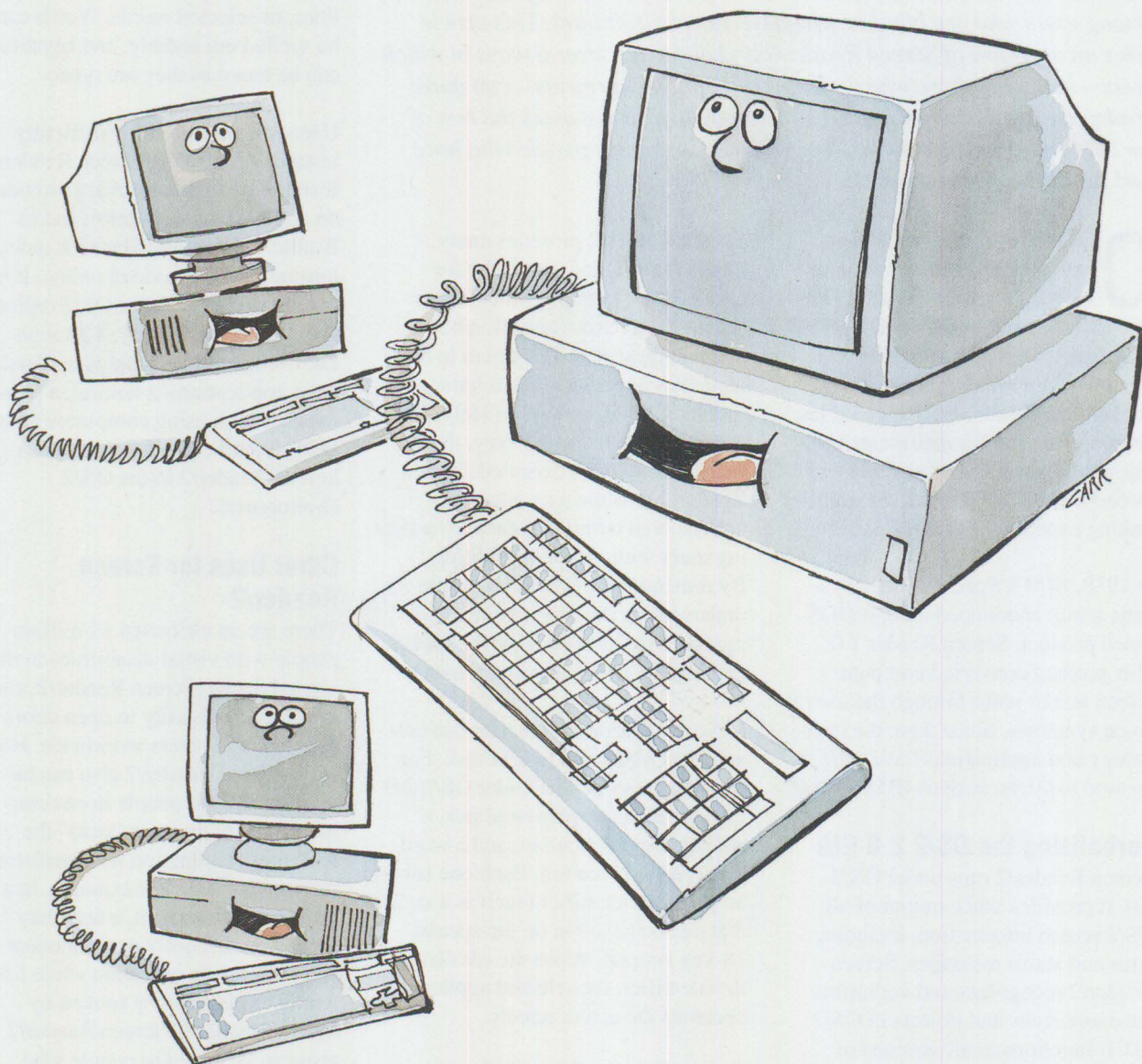
- Raised images of the icons in OS/2 2.0 screens to complement the online tutorial

Screen Reader/2 requires OS/2 2.0 and at least 2 MB of hard disk space. A commercially available text-to-speech synthesizer is also required.

### IBM Independence Series

IBM Screen Reader/2 is one of the products in the IBM Independence

Series. The Independence Series products, which are PS/2-based, provide IBM technology to enhance the employability and quality of life of people who have disabilities. These products are now even more significant because of the Americans with Disabilities Act that went into effect this year. This act requires that employers not discriminate against people with disabilities, and that



employers provide reasonable job accommodations for them.

Independence Series products are available for persons with vision, hearing, mobility, speech, or cognitive disabilities.

### Assistive Accommodations

A new DOS-based version of Screen Reader, IBM Screen Reader/DOS 1.2, enables people who are blind or have visual disabilities to use a computer as a sighted person would. Screen Reader is a software program that converts screen text to speech, so that a user can hear the words on a computer screen and can do word processing, computer programming, and other computer tasks.

IBM PhoneCommunicator™ allows people with hearing or speech impairments to communicate with others who are calling via a standard touch-tone telephone or a Telecommunication Device for the Deaf (TDD). It even provides a full-screen view of the dialog of both parties.

Until now, the ability to use a keyboard was essential for operating computers. Now you can use your voice to operate a computer. IBM VoiceType™ is a DOS-based program for voice-controlled access to, and use of, many text-based programs, including word processors and business applications. VoiceType has a vocabulary of over 7,000 words and is a significant productivity enhancement for persons with upper mobility impairments.

The IBM KeyGuard is a template to enhance keying accuracy on an IBM Enhanced Keyboard or IBM Space Saving Keyboard on PS/2s, AS/400s, RISC System/6000s™, and various IBM terminals. The KeyGuard is a molded keyboard overlay with holes that expose and isolate each keytop. It is appropriate for use by persons with mobility impairments and palsied conditions, and in education and pre-school environments, to enhance keyboarding accuracy.

AccessDOS is an IBM utility that improves and expands access to DOS. It provides extended keyboard, mouse, and sound access on a computer, including StickyKeys, MouseKeys, RepeatKeys, ToggleKeys, and ShowSounds – all important features for people whose mobility is impaired. AccessDOS is available at no charge.

### Clinical Solutions

Designed to increase the efficiency of speech therapy, IBM SpeechViewer™ and IBM SpeechViewer II use the power of the IBM PS/2 to convert elements of speech acoustics to interactive graphic displays that are synchronized with audio playback. DOS-based SpeechViewer includes modules to help develop basic awareness and skill building of selected aspects of speech. SpeechViewer II is DOS- or OS/2-based and includes more advanced modules.

IBM THINKable™ is an OS/2-based clinical tool for computer-assisted cognitive remediation therapy and for attention and memory practice. THINKable provides clinical professionals with an exciting way to offer

skills practice to clients with cognitive disabilities and to enhance case management. THINKable uses colorful, photo-realistic pictures and real speech to help clients practice skills in four critical focus areas: visual attention, visual discrimination, visual memory, and visual sequential memory.

To obtain a brochure about the IBM Independence Series, order G520-9070; for a videotape about the Independence Series, order GV21-9070.

*Ron Hoekzema is a senior marketing programs administrator in the IBM Entry Systems Technology laboratory in Boca Raton, Florida. Ron has held several engineering and engineering management positions and product planning and planning management positions in both hardware and software organizations. He is currently in Special Needs Systems as the marketing programs administrator for Screen Reader/DOS 1.2 and Screen Reader/2.*

*Kathleen Wahl Bode is a senior marketing programs administrator in the IBM Entry Systems Technology laboratory in Boca Raton, Florida. She joined IBM in 1972 as part of its Science Research Associates subsidiary in Chicago. Kathy is currently the Special Needs Systems marketing programs administrator for PS/2-based products for persons with mobility impairments, including VoiceType, AccessDOS, and the KeyGuard.*

# Little Solutions



*We invite you to share your "little solutions" in this column. Send them to us in care of the editor.*

## Icon Editing

When editing or changing an icon in OS/2 2.0, be aware of the icon's bitmap format in OS/2. The following describes the procedure for modifying an icon.

With the right mouse button, choose the icon you want to edit. In the displayed menu, use the left mouse button to select the little arrow next to Open, then select Settings from the next menu. Select the General page from the displayed notebook. Select the Edit pushbutton. The icon editor is now invoked with the current icon of the chosen object. From the icon editor's menu, select Device, then select List from the drop-down menu. A list of bitmap types will be displayed.

Bitmaps are the operating system's representation of the icons that you see. Each bitmap file can contain

many bitmap formats that are used for different displays and purposes. Figure 1 lists all the icon bitmap formats used in OS/2. For example, if you want to change the Desktop icon and you are using a VGA display, select the Independent Color Form (VGA) format.

After making your changes, go to the icon editor's menu. Select File, then select Save from the drop-down menu. Your Desktop icon should now be changed.

Icon Form	Used By
Independent Color Form (VGA)	OS/2 Desktop for VGA displays
8514 (16 colors)	OS/2 Desktop for XGA and 8514/A displays
1 Bit/Pixel (Independent) 16 x 16, 0 x 0, 0 x 0	System icon (the little one in the upper left corner) for VGA displays
1 Bit/Pixel 20 x 20, 1024 x 768, 0 x 0	System icon for XGA and 8514/A displays

Figure 1. Changing the Desktop Icon

## Sorting Out SCSI Adapters

There are differences among the redesigned Small Computer Systems Interface (SCSI) adapters being shipped in PS/2 Models 90 and 95. The differences and a discussion of how to distinguish among these adapters follow.

IBM markets two types of PS/2 Micro Channel SCSI adapters: one with cache and one without cache. There are two versions of the PS/2 Micro Channel SCSI Adapter with Cache: a new one (part number 6451133, feature code 1132) and the old one (part number 6451110, feature code 1018). The 50 MHz models of the PS/2 Models 90 and 95 are shipped with the new SCSI Adapter with Cache. There is only one version of the PS/2 Micro Channel SCSI Adapter without cache (part number 6451109, feature code 1005).

To work properly, a SCSI bus must be terminated on both the internal and external ends of the bus. The SCSI Adapter without cache and the new SCSI Adapter with Cache each have a terminator as part of the adapter. This terminator is a reddish orange resistor located on the top next to the connector end of the adapter.

This terminator can be used to terminate either end of the bus. If there are no external devices on the bus, the terminator becomes the external bus terminator. If there are no inter-

nal devices on the bus, the terminator is used as the internal bus terminator. If both internal and external devices are on the bus, the terminator must be removed. The last device on both ends of the bus also must be terminated with another type of terminator. (Refer to the SCSI device installation instructions for more information.)

The *old* SCSI Adapter with Cache does not have this built-in terminator. It must be terminated differently. If there are no external devices installed, a 60-pin external terminator (part

number 6451039, feature code 1039) must be used. If there are no internal devices installed, a 50-pin terminator (part number 33F8724), which comes with the adapter, must be installed on the top edge of the SCSI adapter. If both internal and external devices are installed, a terminator must be installed on the last device on each end of the bus. (Refer to the SCSI device installation instructions for more information.)

You can distinguish between all these adapters by looking at them. If

the adapter has no SIMM sockets at the upper left end, it is a PS/2 Micro Channel SCSI Adapter without cache. If it has SIMM sockets, it is a SCSI Adapter with Cache. If it is a SCSI Adapter with Cache and it has the reddish orange resistor at the upper left corner, it is the new SCSI Adapter with Cache. If it is a SCSI Adapter with Cache but the reddish orange resistor is not there, it is the old SCSI Adapter with Cache.

— Chuck Hanford, IBM Corporation, Dallas, Texas

## Book Reviews

### *The COBOL Presentation Manager Programming Guide*

One of the current paradigms in personal computer programming is that C language is the only programming language that fully supports OS/2 and the Presentation Manager (PM). The theory that OS/2 and PM programming languages are limited to Macro Assembler or C may have been true historically, but today the correct COBOL compiler can provide as much support for OS/2 or PM as any C compiler. The availability of reentrant COBOL compilers, such as the Micro Focus COBOL/2<sup>®</sup> compiler, allows COBOL programmers to implement the most sophisticated of PM programs while maintaining the familiar, easy-to-understand, source code format that they know.

There remains, however, a significant roadblock for COBOL programmers venturing into PM programming. The lack of COBOL-oriented, technical documentation can slow or even halt the otherwise successful development of COBOL PM applications. Among the volume of OS/2 and PM programming materials, including the materials available from IBM,

there is none that support programming of the Presentation Manager in COBOL. Indeed, within the flood of PM "how-to" programming books, there are none that teach COBOL programmers "how-to."

But, with the recent availability of *The COBOL PM Programming Guide*, the lack of technical documentation has begun to disappear. This book is a combined "how-to" and technical reference book for PM programming, written in COBOL, expressly for use by COBOL programmers.

While providing COBOL programmers with a "fast path" to their first PM window, this first-of-its-kind reference book is also an excellent tool for programmers to begin their PM programming training, regardless of their language specialty. Assuming no prior OS/2 or PM knowledge, the book begins with a complete explanation of the OS/2 and PM environments, moves on to list specific changes that must be made to traditional COBOL structured programs when writing COBOL PM programs, and finally details how to set up a COBOL PM development environment.

Following these introductory topics, *The COBOL PM Programming Guide* lays out, in step-by-step chap-

ters, how to code both the basic PM program and more advanced features such as user dialogs, creating text and graphical output, window subclassing, dynamic linking, multithreading, and device-independent printing.

Special attention has been given to the changes in the traditional COBOL structured programming that programmers face when writing PM programs. The book begins with a discussion of the unique requirements placed on programs by the PM environment. It details the changes in standard COBOL coding that PM requires — everything from the reversed order of the calling stack to how to write reentrant code procedures. For each change required to the traditional structured coding, the book describes the specific code to solve the problem.

While *The COBOL PM Programming Guide* is designed as a teaching manual, it also serves as a technical reference manual for COBOL programmers. Its appendices contain definitions required in every PM program, some of which cannot be found in any other document. The technical references include the following:

- Over 70 pages of sample COBOL PM code

- A skeletal COBOL PM program to form the nucleus of every PM program that is written
- An appendix of the most common PM calls coded in COBOL
- The numerical definitions of over 2,500 PM messages and variables that compose the OS/2 Version 2.0 C header files
- A chart of the 16-bit OS/2 and PM calls, showing how each must change when moving to a 32-bit compiler

By closing the gap in PM's technical documentation, *The COBOL PM Programming Guide* has moved COBOL further along the path toward

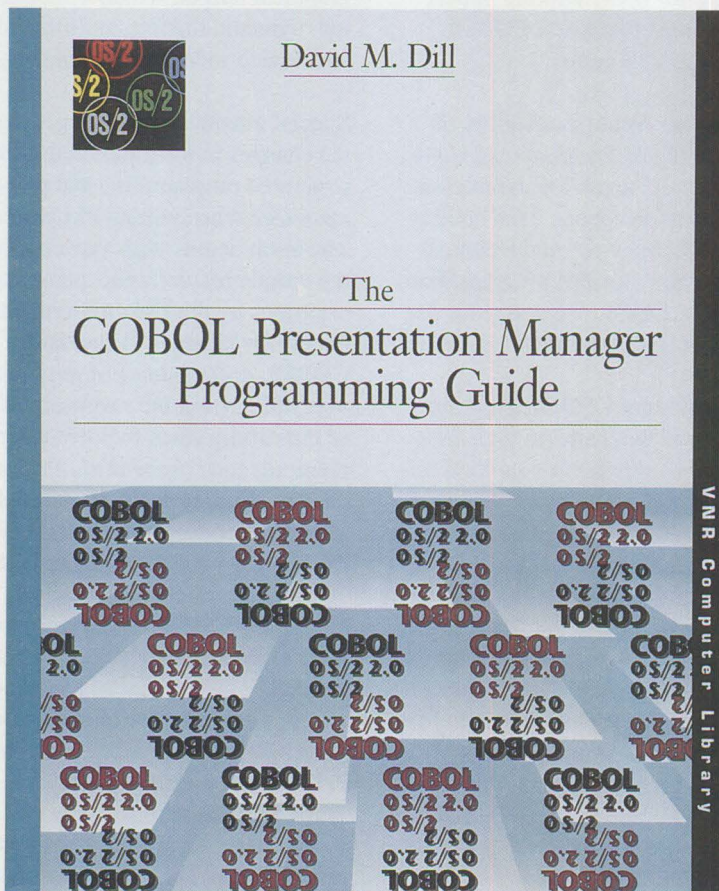
a fully supported PM programming language, and the ability to program PM applications in COBOL can drastically alter the cost equation for a conversion to PM. The traditional COBOL programming shops that cannot justify the move based upon a simultaneous conversion to C and PM can now take another look at the cost and time involved without a C language conversion.

The author, Dave Dill (817 491-2147), provides consulting and education for OS/2 and Micro Focus® COBOL for Presentation Manager. Dave worked for IBM for over 23 years, most recently in customer support, specializing in installations and usage

support for PCs. For the past four years, he provided technical support for the OS/2 Standard Edition operating system, including the OS/2 kernel, Presentation Manager, the OS/2 printing subsystem, the programming toolkit, and COBOL programming under OS/2 and Presentation Manager. Dave was a frequent contributor and technical consultant for *IBM Personal Systems Technical Solutions* magazine.

Dill, David M. *The COBOL Presentation Manager Programming Guide*. Van Nostrand Reinhold: 1992. ISBN 0-442-01293-4. Available from Micro Focus Publishing (800) 551-5269.

IBM form number G362-0010.



### ***Client/Server Programming with OS/2 2.0 (Second Edition)***

Building on the best-selling original edition's strengths, the second edition of *Client/Server Programming with OS/2 2.0* contains over 600 new pages on advanced client/server technology and OS/2 2.0 features. Through easy-to-read, in-depth tutorials and working code, the book shows how to integrate databases, LANs, and Workplace Shell clients to build client/server applications using OS/2 2.0.

The book is written in a friendly style – 100 new cartoons and illustrations – that will appeal to a wide audience ranging from novices to PC programmers and MIS professionals. It also should be of great interest to LAN and communications specialists, database administrators, and anyone with a general interest in client/server technology or OS/2 2.0.

The first 200 pages are virtually a book within a book. Written with mere mortals in mind, this section provides a complete introduction to OS/2 2.0 features, how OS/2 2.0 compares to other client/server platforms (Windows 3.X, UNIX, NT, and NetWare), and OS/2-based client/server

products--including MMPM/2, Database Manager, DDCS/2, Communications Manager, TCP/IP for OS/2, LAN Server, and NetWare Requester for OS/2 2.0. The authors also provide a useful overview of client/server architecture including DCE, multiservers, open systems, and Systems Application Architecture (SAA). This section makes great reading for anyone with a general interest in OS/2 2.0 or client/server technology.

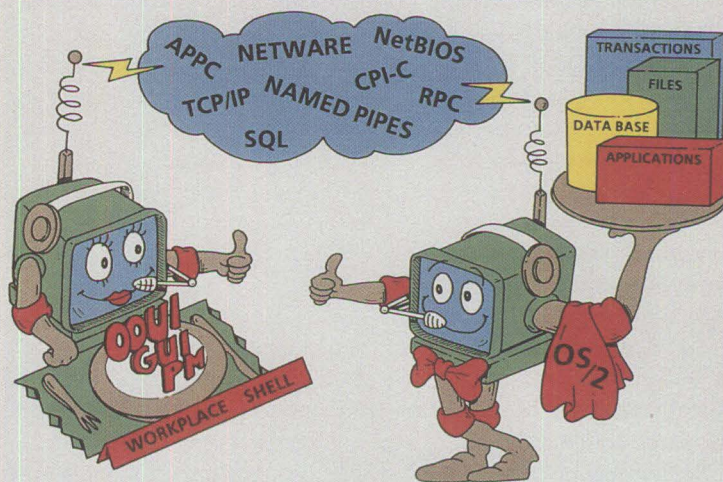
The remaining 900 pages are a programmer's paradise. They cover almost every aspect of programming the OS/2 2.0 client/server platform. The book includes 300 pages of working code using the new 32-bit C Set/2 compiler. All the code is accompanied by detailed tutorials that explain the concepts first. Here's a brief overview of the programming topics covered:

- OS/2 2.0's core services – including threads, 32-bit semaphores, and Named Pipes (200 pages).
- OS/2 2.0's LAN client/server communications, including CPI-C over APPC, NetBIOS, Named Pipes, Sockets over TCP/IP, NetWare, and LAN Server. To make this hard topic fun, the authors stage a BLOB Olympics that compare the performance of these popular protocols (160 pages).
- LAN-based SQL transaction servers using the Extended Services Database Manager and DDCS/2. This includes a detailed introduction to these two database products and to SQL programming. The authors show how to create transaction servers on an OS/2 platform. They use the TP1 benchmark to demonstrate the many trade-offs in client/server design. Which should you use: Static SQL or Dynamic SQL? Remote SQL or Stored Procedures? Transaction Servers or Database Servers? Fat clients or fat servers? You'll know



Robert Orfali  
Dan Harkey

# Client/Server Programming with OS/2<sup>®</sup> 2.0 2<sup>ND</sup> ED.



after reading this section (330 pages).

- Object-Oriented User Interfaces (OOUIs) using PM, SOM, and the Workplace Shell class libraries. The authors explain the benefits of OOUIs and how they can be used to create client frontends. They bring the point home by creating a complete client/server application – a Club Med reservation system. The flamboyant Club Med frontend is created using new objects derived from SOM/WPS classes. The backend uses a transaction server built on top of the OS/2 Database Manager (220 pages).

The book can be used as a supplementary text for courses on OS/2 2.0,

networks, operating systems, Online Transaction Processing (OLTP), LANs, SQL, database theory, OOUIs, and client/server systems. Finally, this is the definitive book for anyone who needs to understand the power OS/2 2.0 offers today as both a client and server platform.

Orfali, Robert and Harkey, Dan. *Client/Server Programming with OS/2 2.0 (Second Edition)*. Van Nostrand Reinhold: 1992. ISBN: 0-442-01219-5. Available at local bookstores and from the publisher (800) 842-3636.

IBM form number G325-0650.

# New Products

## IBM PS/note Model N45 SL Notebook Computer

The IBM PS/note Model N45 SL is a high-performance, lightweight, battery-operated (AC/DC) notebook PC. Standard features include a 25 MHz 80386 SL processor with 64 KB cache, 2 MB of 80 ns memory (expandable to 8 MB), 2.5-inch 80 MB or 120 MB hard disk, 3.5-inch 1.44 MB diskette drive, 10-inch 32 grayscale Video Graphics Array (VGA) resolution LCD, 82/83-key keyboard, serial port, parallel port, keyboard/mouse port, external VGA port, rechargeable battery pack, and AC adapter.

Also announced as options for the IBM PS/note Model N45 SL are 2 MB RAM, 2400 baud data modem, international fax/data modem (2400 baud data, 9600 baud fax), IBM PS/note NiCd Battery for Model N45 SL with a battery cradle for quick charging, PS/2 miniature mouse, and the IBM PS/note AC Adapter for Model N45 SL. The PS/note carrying case is available as an accessory.

International Hardware Warranty Service is offered for users who travel worldwide to countries where this product is sold by IBM, IBM Authorized Personal Computer Dealers, or IBM Authorized Industry Remarketers.

### Highlights:

- Large, clear, and vivid liquid crystal display
- A high-performance processor (25 MHz 80386 SL), expandable memory (2 MB to 8 MB), large

hard disk (80 MB or 120 MB), and long battery life

- Rugged, soft-touch, and scratch-resistant finish

*Letter # 192-162, July 21, 1992*

## IBM PS/2 Model 57 SLC

PS/2 Model 57 SLC complements and enhances the PS/2 family of Micro Channel systems with the integration of the IBM 386 SLC 20 MHz microprocessor that includes 8 KB of internal cache and the internal cache controller. Also included are a high-performance, large hard disk; a 2.88 MB media sense diskette drive; a 16-bit VGA adapter; a Small Computer Systems Interface (SCSI); and 8 MB of memory housed in the same versatile 5-slot/4-bay mechanical package as other IBM PS/2 57 SX and 57 SLC systems. Model 57 SLC is offered with a disk capacity of 400 MB.

The system allows horizontal or vertical operation and ships with a stand for the vertical orientation. Five 16-bit Micro Channel slots allow users to expand and customize system functions. A hard disk and diskette drive are installed in each system, and two expansion bays are available to accept additional 5.25-inch and 3.5-inch diskette drives, hard disks, tape, CD-ROM, or other devices. The SCSI controller is integrated into the planar, so no slot is required to attach up to seven SCSI devices (including the standard hard disk). Two additional SCSI devices can be installed internally and the remainder externally with the external connector.

PS/2 Model 57 SLC comes with two 4 MB, 70 ns Single In-Line Memory

Modules (SIMMs) and allows installation up to a total of 16 MB of 70 ns memory on the planar. The memory controller automatically supports enhanced performance through memory interleaving with appropriate memory configurations. The standard video for the PS/2 Model 57 SLC is 16-bit VGA.

The IBM PS/2 Model 57 SLC supports the IBM Enhanced Keyboard (101-key), Space Saving Keyboard (84-key), IBM Host-Connected Keyboard (122-key), and a variety of languages.

*Letter # 192-177, August 11, 1992*

## IBM PS/2 486DX2-66 Processor Upgrade Option

The IBM PS/2 486DX2-66 Processor Upgrade Option enhances the PS/2 Model 90 XP 486 and Model 95 XP 486 system families. This new upgrade option and its Intel 486DX2-66 microprocessor provide the capability to upgrade processor performance on the PS/2 Model 90 XP 486 and Model 95 XP 486 systems.

The 486DX2-66 microprocessor operates at an internal clock speed of 66 MHz and an external speed of 33 MHz. It includes an internal memory cache controller, 8 KB memory cache, and an integrated floating-point processor unit. Additionally, the processor complex supports the external PS/2 256 KB Level 2 Cache Option.

The 486DX2-66 Processor Upgrade Option significantly increases system processor performance in compute-intensive applications, providing investment protection to owners of lower performance models of the PS/2 Model 90 XP 486 and Model 95 XP 486 systems.

*Letter # 192-178, August 11, 1992*



### **IBM PS/2 8511 Color Display Model 001**

The PS/2 8511 Color Display Model 001 is a 14-inch, medium-resolution VGA color display with smaller dot pitch and sharper characters relative to the IBM PS/2 8512 Color Display. The 8511 Color Display, a derivative of the 8518 Color Display, is a suitable display for VGA users. The low emissions standards, Extremely Low Frequency Magnetic Field (ELMF), and Very Low Frequency Magnetic Field (VLMF) are in response to user requests and emerging international requirements.

*Letter # 192-180, August 11, 1992*

### **IBM 4070 IJ Printer**

The IBM 4070 IJ Printer is a small, letter-quality, ink-jet printer designed as a low-usage, personal desktop printer. The 4070 IJ prints at up to 83 characters per second (cps) in letter-quality mode and up to 110 cps in high-speed mode on plain paper at a quiet 45 dBA level.

There are two models of the 4070 IJ: Model 001 includes a 50-sheet Automatic Sheet Feeder (ASF), and Model 002 offers the ASF as an option. An optional battery pack is available for both models.

*Letter # 192-176, August 11, 1992*

### **Selected IBM PS/2 Models Upgraded to 8 MB of Standard Memory**

IBM has enhanced standard system memory to 8 MB on PS/2 Model 56 SLC, Model 57 SLC, and Model M57 SLC systems. With this change, all enhanced systems have double the standard system memory to better address user application requirements.

The PS/2 Model 56 SLC, Model 57 SLC, and Model M57 SLC systems include OS/2 2.0. A 160 MB hard disk and OS/2 2.0 combine for a functionally adaptive, high-performance

desktop system with a hardware platform conducive to solutions for a range of user requirements.

These selected systems are shipped with two 4 MB, 70 ns SIMMs and allow up to 16 MB 70 ns memory on the planar. The memory controller automatically supports enhanced performance through memory interleaving with the appropriate memory configurations.

*Letter # 192-165, July 21, 1992*

### **IBM PS/2 Micro Channel to Mainframe Connection**

The IBM PS/2 Micro Channel to Mainframe Connection, when used with appropriate software, enables users to connect an IBM Micro Channel PS/2 directly to a System/370™, System/390®, ES/3090™, or ES/9000™ channel running in block-multiplexer mode. Since no communications protocol is involved, the transfer of data between the two systems can be up to 4.5 MB per second.

Attachment to ESCON™ channels is supported by the IBM 9034 ESCON Converter Model 1. This adapter replaces the IBM PS/2 System/370 Channel Adapter.

#### **Highlights:**

- Intelligent adapter with onboard microprocessor and data buffers.
- Up to two Micro Channel-to-Mainframe Connection adapters can be installed in a single PS/2 computer.
- Up to three Micro Channel-to-Mainframe Connection adapters can be daisy-chained on the same host system channel.

*Letter # 192-156, June 30, 1992*

### **IBM 5183 Portable Printer**

The IBM 5183 Portable Printer is a compact, letter-quality, plain paper,

thermal transfer printer designed to fit in a briefcase along with a laptop or notebook computer. The printer is battery-powered and prints in letter-quality at up to 53 characters per second.

#### **Highlights:**

- Lightweight (only 2.5 pounds)
- Letter-quality printing
- Eight-inch print line across page
- Battery-powered operation
- Standard paper or transparencies
- 4 KB integrated buffer

*Letter # 192-167, July 28, 1992*

### **LAN Connection for Printers in Novell NetWare Environment**

The IBM LAN Connection for Printers and Plotters provides a direct connection of parallel and RS232C serial printers and plotters to Local Area Networks (LANs). These support Novell NetWare networks. The IBM LAN Connection for Printers and Plotters consists of a network printer adapter unit that attaches to a LAN, and a software utility that resides on the LAN print server. The adapter unit has both a serial and a parallel port that can be used simultaneously to support two printers or a printer and a plotter.

The software utility takes the print data normally sent to the print server's parallel ports and redirects it to either the parallel or serial port of the adapter. The appropriate printer attached to the adapter receives the data and prints it just as if the printer were attached to the server's printer port.

The IBM LAN Connection for Printers and Plotters software utility supports remote printing in PSERVER or RPRINTER mode with Novell NetWare 2.2 or 3.11 connected to either a Token-Ring (IEEE 802.5) or

an Ethernet (IEEE 802.3 or Ethernet II) LAN.

The IBM LAN Connection for Printers and Plotters operates without noticeable degradation of printing performance and supports the following data streams: IBM Personal Printer Data Stream (PPDS), Hewlett-Packard® Printer Control Language (HP PCL), IBM/HP Graphics Library (GL), and Adobe PostScript®. A friendly interface allows users to easily configure and maintain one or more adapters on a network.

#### Highlights:

- PSERVER and RPRINTER support is provided.
- The 4033 provides simultaneous support for connecting printers or plotters:
  - One parallel printer
  - One RS232C serial printer or plotter port
- The 4033 Model 011 provides support for Token-Ring 4/16 Mbits/second data rate (IEEE 802.5).
- The 4033 Model 012 provides support for Ethernet 10BaseT 10 Mbits/second data rate (IEEE 802.3).
- The 4033 Model 013 provides support for Ethernet 10Base2 and 10Base5 10 Mbits/second data rate (IEEE 802.3 or Ethernet II).
- Printers and plotters can be located in user work areas, which can be remote from the print server.
- Eight file servers, two preferred file servers, and thirty-two print queues are supported.
- The 4033 is compatible with IBM PPDS, Adobe PostScript, HP® PCL, and IBM/HP GL.

*Letter # 192-173, July 28, 1992*

#### IBM PS/2 Adapter/A for Ethernet Twisted-Pair Networks

IBM enhances its LAN family of products by announcing the Personal System/2 Adapter/A for Ethernet twisted-pair networks. This LAN adapter provides the ability to attach PS/2 Micro Channel bus systems to Ethernet networks. The adapter supports Ethernet Version 2 and IEEE 802.3 CSMA/CD with data transfers at 10 Mbits/second through an Attachment Unit Interface (AUI) or RJ45 connector. Connection to 10BaseT (unshielded twisted-pair cable) is provided through the RJ45 connector. Connection to 10Base2 (thin coaxial cable) or 10Base5 (thick coaxial cable) is provided through the AUI connector and a user-provided external transceiver. Remote Program Load (RPL) is included as standard for systems requiring RPL capability.

#### Highlights:

- Offers both Ethernet and token-ring products to create intelligent LANs
- Allows Ethernet/802.3 users to expand networks
- Supports multiple interfaces and LAN software
- Supports Ethernet Version 2 and IEEE 802.3 CSMA/CD interfaces
- Provides an easy-to-use interface for installation and setup, configuration and diagnostic capabilities, and maintenance

*Letter # 192-158, June 30, 1992*

#### LAN Automated Distribution/2 Service

LAN Automated Distribution/2 (LAD/2) Service, a system-engineer-delivered billable service, enables the installation and configuration of OS/2 2.0, OS/2 EE 1.3, and DOS from a distribution server to PS/2s on IBM Token-Ring and Ethernet LANs.

With LAD/2, you also can migrate from:

- OS/2 1.2 or 1.3 to OS/2 2.0 (Base)
- IBM DOS 3.3 and above or DOS with Windows to OS/2 2.0
- OS/2 EE 1.2 to OS/2 EE 1.30.2
- IBM DOS to Dual Boot

IBM system engineers who have been qualified on LAD/2 can perform a billable service for users who want to install/distribute the following:

- IBM OS/2 2.0
- IBM Extended Services
- IBM LAN Server 2.0 (LAN Requester only)
- IBM OS/2 Extended Edition 1.30.2
- IBM DOS 5.0
- IBM Personal Communications/3270
- IBM LAN Support Program
- IBM DOS LAN Requester
- IBM DOS Database Requester
- Microsoft Windows
- Novell NetWare Requester
- OS/2, DOS, Windows, and user-developed application software

For more information about this service, call (800) 547-1283.

#### IBM Education LAN and Tools 386

IBM Education LAN and Tools 386 (EdLAN 386) is an education network package of network software, productivity tools, and related publications. The network software includes IBM Classroom LAN Administration System Version 1.40 and NetWare from IBM V3.11. The productivity tools consist of:

- LinkWay™ Version 2.01
- Microsoft Works Version 2.0A
- LANSchool® Version 3.2

- Excelsior grade2 Version 1.0
- Excelsior quiz2 Version 1.0
- Express Publisher™ Version 2.0

EdLAN 386 is the same as IBM's existing EdLAN Version 1.10 network package, except EdLAN 386 includes NetWare from IBM V3.11 in place of V2.2. EdLAN Version 1.10 continues to be available. LANSchool Version 3.2 is the latest release of LANSchool; however, the publication content for the present version has not changed.

EdLAN 386 (100 User) enables physical access to IBM Classroom LAN Administration System Version 1.40 and NetWare from IBM V3.11 by up to 100 workstations concurrently. Use of each of the productivity tool programs included in EdLAN 386 (50 and 100 User) is limited by its license terms to a maximum of 50 machines at any one time (except for LANSchool).

#### Highlights:

- NetWare from IBM V3.11 enhancements include increased performance and protection. The installation utility greatly improves the setup time over previous versions of NetWare. Startup configuration can be altered easily without going through a lengthy reconfiguration process.
- Systems management through the use of IBM Classroom LAN Administration System Version 1.40 helps reduce complexity and improve productivity for courseware and other applications management.
- Comprehensive productivity tools include a gradebook, test generator, spreadsheet, desktop publishing capability, word processor, database, and workstation controller.
- IBM LinkWay Version 2.01 enables teachers and students to

design and create text, pictures, and graphics presentations.

*Letter # 292-348, June 16, 1992*

#### IBM ActionMedia II Developer's Toolkit Version 1.1

The ActionMedia™ II Developer's Toolkit Version 1.1 allows application and tool developers to become more productive in creating ActionMedia II programs. The Developer's Toolkit provides the same function as Version 1.0, with the following additional features:

- Executes with Audio-Video Kernel 1.1
- Executes in an OS/2 Version 2.0 operating environment
- Executes in a Windows 3.1 environment
- Provides sample C source code for use in a Windows 3.1 environment

The Developer's Toolkit also includes the *ActionMedia II Technical Reference* manual and a programmer's guide.

The ActionMedia II Developer's Toolkit Version 1.1 Upgrade Kit gives users of ActionMedia II Developer's Toolkit Version 1.0 the capability to upgrade to Version 1.1 without any additional charge.

#### Highlights:

- Source C code and a programmer's guide for use by developers to acquire skills in using the ActionMedia II Audio-Video Kernel programming interface
- File utilities for developers to edit and check the Audio/Video Support System (AVSS) files created

*Letter # 292-360, June 23, 1992*

#### IBM PS/2 ActionMedia II Upgrade Kits

IBM announces enhancements to the ActionMedia II display adapters and offers two ActionMedia II upgrade

kits. The ActionMedia II display adapters include enhanced device drivers and programming libraries that support a new ActionMedia II Audio-Video Kernel programming interface. Audio-Video Kernel Version 1.1 is provided for use with OS/2 Versions 1.3 and 2.0, and Windows 3.1. These ActionMedia II multimedia options, part of the UltraMedia™ product family from IBM, allow the full range of still natural images, motion video, and quality audio information to be incorporated into new PS/2 multimedia applications.

The ActionMedia II Upgrade Kit 1 includes enhanced device drivers (Audio-Video Kernel Version 1.1) for use with OS/2 Version 1.3 and Windows 3.1. This upgrade kit is intended for users who previously purchased the ActionMedia II Display Adapter/A (2 MB) and are using the Audio-Video Kernel Version 1.0.

The ActionMedia II Upgrade Kit 2 includes enhanced device drivers (Audio-Video Kernel Version 1.1) and Media Control Interface programming libraries for use with OS/2, MPM/2, and Windows 3.1. This upgrade kit is intended for users who previously purchased either of the ActionMedia II display adapters and require upgrading to an OS/2 2.0 environment, or who wish to use the Media Control Interface programming libraries for Windows 3.1.

*Letter # 192-153, June 23, 1992*

#### IBM LAN Distributed Platform Programs

The IBM LAN Distributed Platform (LANDP) products provide a client/server distributed programming capability well suited for applications in a LAN environment of mixed operating systems (DOS, OS/2, and AIX®) and mixed machine types (PCs, PS/2s, and RISC System/6000s). Clients, servers, or a combination of clients and/or servers can reside in any

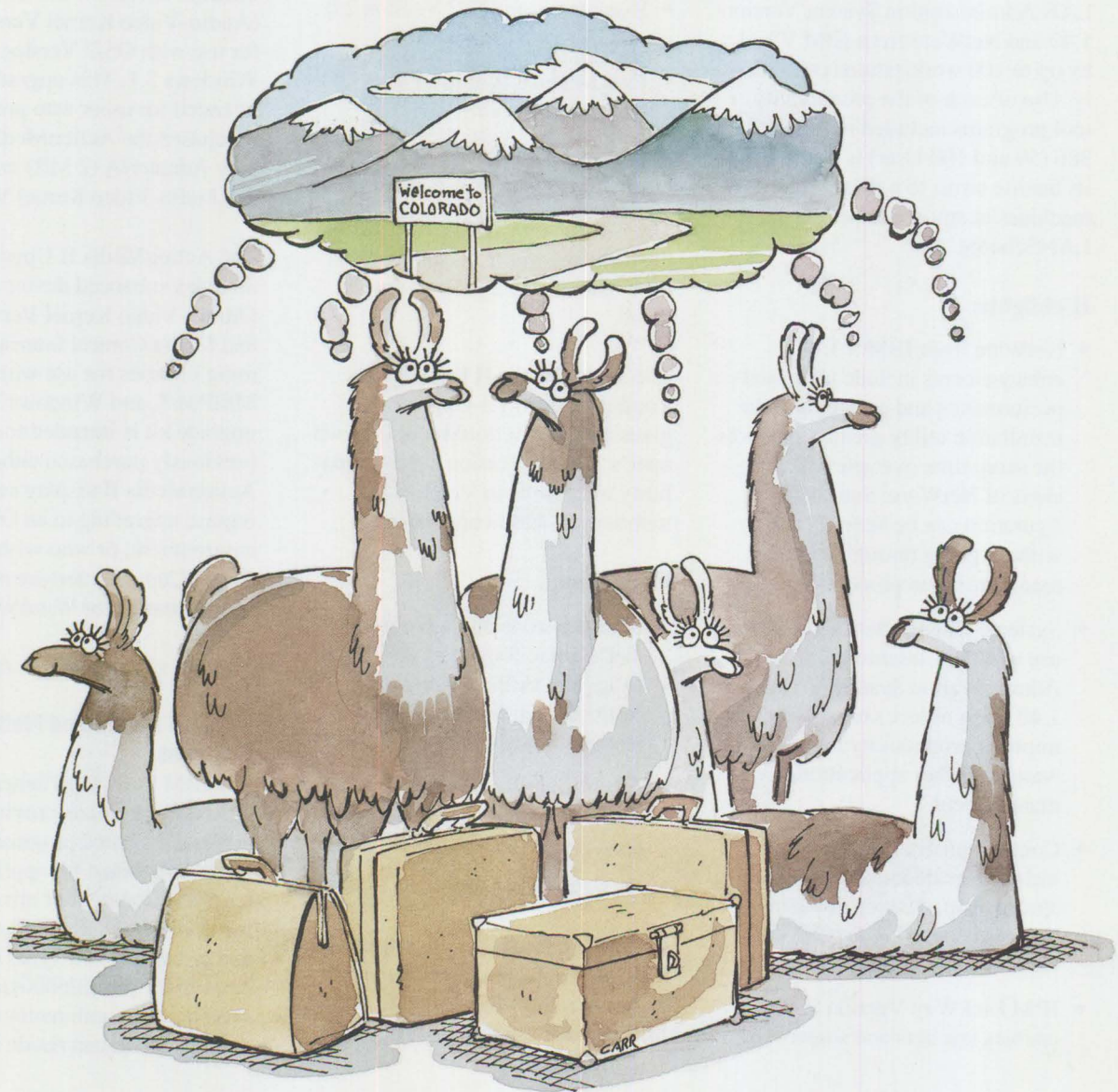
machine. A common Application Programming Interface (API) across platforms and an open design that accommodates user-written servers also are provided. LANDP/DOS and LANDP/2 are introduced with this announcement. In the OS/2 environment, LANDP/2 provides enhanced services built upon the OS/2 LAN Server, OS/2 LAN Requester, OS/2 Communications Manager, and OS/2 Database Manager, through servers and a high-level LANDP call API.

Additionally, LANDP products provide a significant set of functions, application services, and servers to facilitate resource sharing, communications, systems management, software distribution, and other system and general-purpose functions.

LANDP evolves from the IBM Financial Branch Systems Services (FBSS) family of products and extends and enhances the functionality provided by FBSS products. In addition,

LANDP/2 extends the FBSS/2 Version 1.1 functionality from 16-bit implementation to 32-bit implementation for support of OS/2 Version 2.0. Mixed LANs including FBSS and LANDP workstations are supported, and upward compatibility for FBSS applications is provided.

*Letter # 292-362, June 23, 1992*



Q

Where do users get information on AIX and the RISC System/6000?

A

They go straight to the source, and so should you...

# /AIXtra

If you use AIX or have an interest in open systems, you could use a single, comprehensive source of information devoted to your needs. /AIXtra: IBM's Magazine For AIX Professionals is that resource. Whether you're an experienced system administrator or you're just starting to use UNIX, there are many reasons to read /AIXtra:

- ◆ Stay informed about the latest products for AIX and the RISC System/6000.
- ◆ Enjoy many well-written, educational articles about subjects like networking, communications, system administration, graphical user interfaces, distributed computing, and much more.
- ◆ Learn how others have harnessed the power of AIX and the RISC System/6000.
- ◆ Check the /AIXtra Field Television Network (FTN) schedule for monthly satellite broadcasts just for AIX users.
- ◆ Get detailed information about exciting trade shows and expositions.

There are many reasons to read and enjoy /AIXtra, and there also are many ways to subscribe. Choose a subscription method that is most convenient for you. Then either copy this page and fax it to (415) 948-4280, or call The TDA Group at (800) 551-2832 and request your subscription. Another option is to write to TDA at P.O. Box 1360, Los Altos, CA 94023. Please provide the following information:

- 1 Year Subscription (4 issues) for \$35\*
- 2 Year Subscription (8 issues) for \$64\*
- 1 Year Subscription (surface delivery) Canada/Mexico \$60†; other countries \$80†
- Charge to my VISA/MasterCard
- Bill me Purchase Order# \_\_\_\_\_
- Check enclosed (Make checks payable to The TDA Group)

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

City/State/Zip: \_\_\_\_\_ Expires: \_\_\_\_/\_\_\_\_/\_\_\_\_

VISA/MasterCard #: \_\_\_\_\_

Phone: \_\_\_\_\_

\*California residents add applicable sales tax.  
 †Canada/Mexico and other international subscriptions must be prepaid in U.S. dollars.



# FOCUS ON MICRO CHANNEL ARCHITECTURE

Here's a valuable reference on Micro Channel architecture. We've collected articles from past issues of *IBM Personal Systems Technical Solutions* and other sources and reprinted them in a single publication.

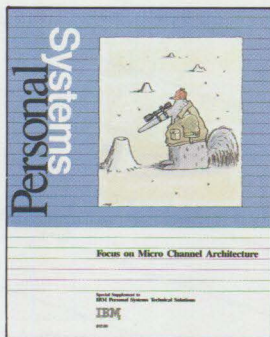
*Focus on Micro Channel Architecture* provides in-depth technical explanations of the architecture of IBM's PS/2 Micro Channel systems. A foreword by Ramiz Zakhariya, president of the Micro Channel Developers Association, is testimony to the support of industry leaders.



Yes...

Send me \_\_\_\_\_ copies of *Focus on Micro Channel Architecture* at \$12.00\* plus \$3.95 shipping and handling, a total of \$15.95\*\* per copy.

\*Discounts available for multiple copies.  
\*\*California residents add applicable sales tax.



Charge to my VISA/MasterCard       Check       Money Order

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

City/State/Zip: \_\_\_\_\_

Phone: \_\_\_\_\_

VISA/MasterCard #: \_\_\_\_\_ Expires: \_\_\_\_\_/\_\_\_\_\_/\_\_\_\_\_

Make checks payable to: The TDA Group, P.O. Box 1360, Los Altos, CA 94023. For faster service, order toll-free by calling (800) 551-2832, or fax this form to (415) 948-4280. All orders must be prepaid.

## Index to Past Issues of IBM Personal Systems Technical Solutions

### July 1992 (G325-5017)

IBM PS/2 Server 295: New Thresholds for Client/Server Networking  
Comparing Architectures: Micro Channel and EISA (Part 2)  
Synergy by Design  
Pen-Based Computers  
Why Doesn't My Portable's Battery Last Longer?  
Planning Guidelines for Token-Ring Cabling  
Installing and Migrating Applications in OS/2 2.0  
Printing Under OS/2 2.0  
Installing the IBM 4029 LaserPrinter Under OS/2 1.3  
Serviceability Tools in OS/2 2.0  
Online Communication Using the OS/2 2.0 PM Terminal  
IBM Extended Services Database Manager  
NetWare for SAA  
Using the IBM DOS 5.0 Driver EMM386.EXE and Upper Memory  
The Solutions Evaluation Tool

### April 1992 (G325-5015)

Comparing Architectures: Micro Channel and EISA  
Portable Computer Trends and Directions  
LCD Panel Technology  
The OS/2 Workplace Shell  
New Applications in OS/2 2.0  
Unattended Installation of OS/2 2.0  
OS/2 Communications Manager Trace Events  
IBM and Novell LAN Software Coexistence  
IBM 8209 LAN Bridge Connects Ethernet Clients to Novell and IBM Servers  
Backup and Restore in an IBM NetWare Environment  
The DOS Protected-Mode Environment  
DOS Disk Management  
Customizing Alphanumeric Screen Dimensions

### January 1992 (G325-5014)

Additions to the IBM PS/1 Family  
IBM LaserPrinter 4029 Series Print Quality Enhancements  
OS/2 2.0: The Integrating Platform  
Multiple Virtual DOS Machines  
IBM OS/2 LAN Server 2.0  
OS/2 2.0 Memory Management  
Coding for Performance Under OS/2 Version 2.0  
Extending the Functions of OS/2 REXX  
Protecting User Exits Under OS/2 1.X  
GDDM-OS/2 Link  
IBM Upgrade Enhanced Install Utility/DOS 5.0  
Advanced Peer-to-Peer Networking: An Overview  
Using IBM SAA Networking Services/2  
The AAI Family of Products  
Securing the Enterprise Workstation

### Issue 4, 1991 (G325-5013)

Power Factor: Non-Linear Loads and the Power Distribution System  
Database Manager: Highlights and Direction  
OS/2 Communications Manager  
Improving OS/2 Application Performance  
Creating PM Windows with Dialog Templates  
REXX Program for OS/2 LAN Server  
Micro Focus COBOL/2 and the DOS Database Requester  
IBM DOS 5.0 Facts and Features  
IBM DOS 5.0 Upgrade  
DOS 5.0 Performance Improvements  
DOS Memory Management Facilities  
Disk Caching Under DOS  
NetWare Client-Server Interaction  
LANACS Protocols

### Issue 3, 1991 (G325-5012)

PS/2 Model L40 SX Laptop Portable Computer  
OS/2 2.0 Considerations  
Comparing PC-DOS, OS/2, and AIX PS/2 – Part 2  
Using Dual Displays with OS/2  
Local Area Networks: The New Utility  
And You Thought LANs Were Just for the Office!  
Remote LAN Management Tools

New Horizons for IBM's Shielded Twisted Pair Cabling  
Tuning and Self-Tuning Features of OS/2 LAN Server  
NetWare Communications and Routing Protocols  
Little Solutions for LANs

### Issue 2, 1991 (G325-5011)

IBM PS/2 Model 90 XP 486 and Model 95 XP 486  
Choosing an I/O Bus Architecture  
The Network Is the Message  
Invoking Printer Job Properties  
Comparing PC-DOS, OS/2, and AIX PS/2  
Programming PM Using the COBOL/2 Bindings  
Installing and Using the DOS Database Requester  
OS/2 LAN Server 1.3 Overview  
IBM Windows Connection 2.0  
SNA Definitions for 3270 Emulators – Part II  
IBM THINKable™

### Issue 1, 1991 (G325-5010)

XGA – Raising Video Expectations  
Choosing betw. Shielded and Unshielded Wiring for Data Transmission  
Compatibility of LAN Servers and Requesters  
Running DOS LAN Requester and Novell NetWare Concurrently  
Breaking the 640 KB DOS Memory Barrier  
Understanding an OS/2 CONFIG.SYS File  
OS/2 EE 1.2 Database Manager Performance  
OS/2 EE 1.2 Competitive Performance  
An Intelligent Front-End EASEL® Application  
Enabling Software for National Language Support  
SNA Definitions for 3270 Emulators  
Diskette Failures Caused by Contamination

### Issue 4, 1990 (G325-5009)

First Look at the New IBM PS/1 Computer  
Using the IBM 4019 LaserPrinter Effectively  
Micro Channel Interface Chip Sets  
Token Ring Bus Master LAN Adapters  
Extension of Wiring Rules for 4-Mbit/s Token Ring Using UTP Lobes  
SCSI and DISK.386.SYS  
Operating System Platforms: A Business Perspective  
Minimum OS/2 1.2 DASD Requirements  
User Profile Management  
Understanding OS/2 1.2 LAN Server Performance  
PM: An Object-Oriented Approach  
DOS 4.00 SHARE  
A "C" Programming Model for DOS Device Drivers  
An Electronic Bulletin Board for PC Users

### Issue 3, 1990 (G325-5007)

DOS – A Look Under the Hood to See How It Spins  
Memory Management in a DOS Environment  
FASTOPEN – The DOS Performance Enhancer  
DOS 4.00 Compatibility Issues  
'Out of Environment Space' Errors  
A New LAN Requester for DOS Systems  
Creating a Dialog Box Dynamically Using WinCreateDlg  
An Alternative for the OS/2 START Command  
CUA: A Consistent Interface

### Issue 2, 1990 (G325-5006)

OS/2 End User Advantages  
What's New in OS/2 Standard Edition Version 1.2?  
An Application Developer's View of OS/2  
Object-Oriented Programming with C and OS/2 PM – Is It Possible?  
Design Goals and Implementation of the New HPFS  
OS/2 EE 1.2 Database Manager – Remote Data Services  
OS/2 EE Database Manager Precompiler API  
UNION, INTERSECT, EXCEPT  
Writing a Database Manager COBOL/2 Program  
Database Manager Programming with Procedures Language 2/REXX  
APPC Performance Tips for OS/2 EE  
EASEL OS/2 EE PROFS: Host Code Interface  
PS/2 RPG II Application Platform and Toolkit  
The IBM Independence Series Products

“High-performance servers typically employ busmaster network and disk adapters. (page 2)

“With the IBM video monitoring solution, organizations can broadcast live to workers at their desks. (page 18)

“Why is it impossible to access some physical memory? What happens to it? How is it used? (page 21)

“The OS/2 Boot Manager enables you to have multiple operating systems installed in a computer. (page 27)

“New APIs in OS/2 2.0 support 32-bit applications. (page 42)

“The *environment* of an OS/2 session is a special area in storage that contains information specific to that OS/2 session. (page 56)

“Micro Channel adapter configuration is easier than EISA adapter configuration. (page 69)

“Sensitivity studies show the effects of tuning some parameters. (page 71)

“AlertVIEW detects application, operating system, and communication software problems when they occur. (page 82)

“Screen Reader/2 is the first tool that makes IBM's Graphical User Interface (GUI) available to blind users. (page 85)

G325-5019-00

