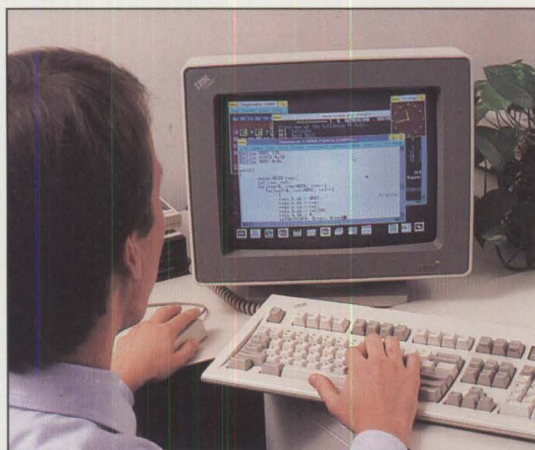
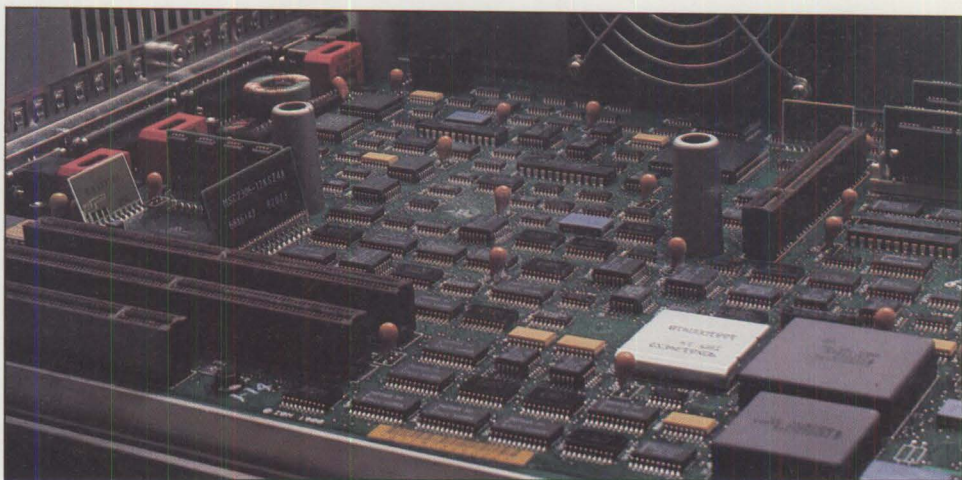
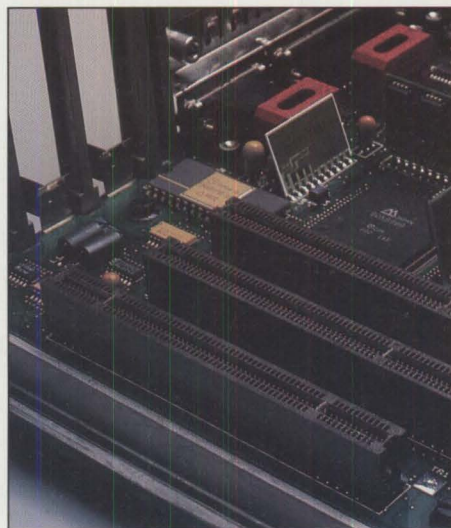


Issue 4, 1989

Personal Systems



**SPECIAL
MICRO
CHANNEL™
ISSUE**



IBM Personal Systems Technical Journal



IBM Personal Systems Technical Journal is published by the U.S. Marketing and Services Group, International Business Machines Corporation, Roanoke, Texas, U.S.A.

Editor	Libby Boyd
Consulting Editor	Mike Engleberg
Technical Consultant	Joel Cohen
Production Coordinator	Leigh Bingham
Automation Consultant	Andrew Frankford
Illustrator	Bill Carr
Manager	Gene Barlow

To correspond with the *IBM Personal Systems Technical Journal*, please write to the Editor at

IBM Corporation
Internal Zip 40-C2-02
One East Kirkwood Blvd.
Roanoke, TX 76299-0015

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

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

Titles and abstracts, but no other portions, of information in this publication may be copied and distributed by computer-based and other information-service systems. Permission to republish information from this publication in any other publication or computer-based information system must be obtained from the Editor.

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

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

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

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

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

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

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

All specifications are subject to change without notice.

Copyright © 1989 by International Business Machines Corporation

THANK YOU

Thanks to Bob Lohman, Manager, Special Marketing Programs, who supported our efforts and whose staff contributed to this Micro Channel issue.

My special thanks to Joel Cohen, who provided technical direction and coordination for this issue and who was committed to producing information that will be of value to our readers.

Joel J. Cohen is in Special Marketing Programs in IBM's Entry Systems Division at the Boca Raton Laboratory. His responsibilities include the development and coordination of PS/2 demonstrations at business shows. His experience in the computer field extends from centralized batch-oriented to online data processing to personal systems. In a period of over 30 years he has held a variety of staff and management positions involving the design, development, marketing, assurance, usability, and marketplace evaluation of computers. Joel joined IBM in 1964. He holds a bachelor of science degree in electrical engineering from the University of City College in New York and a master of science degree in electrical engineering from Northeastern University in Boston.

Editor

Contents

Hardware

Micro Channel Architecture

- 1** Executive Comments
- 3** Requirements for Advanced Bus Architecture
- 23** Micro Channel System Configuration Considerations
- 43** Features and Benefits
- 48** Bus Masters and Applications

Extended Features

- 53** Overview of Extended Micro Channel Functions
- 61** New Micro Channel Features
- 66** SCB - An Architecture for Micro Channel Bus Masters

Bus Master Design and Applications

- 83** Design Alternatives with Micro Channel Systems
- 86** IBM PS/2® Micro Channel SCSI Adapters
- 93** PS/2 Wizard Adapter
- 96** Experience in Bus Master Design
- 99** Bus Master Adapters from Independent Option Vendors

Software

- 104** Bus Masters and OS/2 ®
- 112** Micro Channel Issues in AIX™ PS/2

Random Data

- 119** Information for Developers
- 124** Book Report
- 125** New Products

Executive Comments

*Dr. Robert L. Carberry
IBM Corporation
Somers, New York*

This special issue of the *IBM Personal Systems Technical Journal* highlights IBM's technical leadership and innovation in the definition, design, development and use of Micro Channel architecture.

Micro Channel architecture has rapidly established itself as the new industry standard. More than half of the IBM Personal System/2® computers being shipped today are Micro Channel-based, a trend that is continuing to accelerate. As of August 1989, there were over 1000 Micro Channel card products from over 300 independent developers. There were already 27 advanced-function bus master cards available. And, 197 Micro Channel card IDs have been issued to 93 companies for additional bus master cards – an indicator of much more to follow. Twenty-two other companies have acknowledged the significance of Micro Channel architecture by announcing Micro Channel-compatible systems.

In September of this year, IBM released additional information about Micro Channel architecture, which expands performance, improves system integrity and enhances programmability. The basic I/O data transfer capability of the Micro Channel on the Personal System/2



is up to 20 million bytes per second today. This information describes two additional modes that permit data transfers of up to 40 million bytes per second and 80 million bytes per second, with extendability to 160 million bytes per second. Additional fault detection and isolation capability is provided with data parity, address parity and synchronous channel-check features. Pro-

grammability is expected to be enhanced by the introduction of Subsystem Control Block architecture, a standard for communications between bus masters.

In making this announcement, we have demonstrated our continuing commitment to an architecture that provides a sophisticated and a broad platform for multiple families of



products, while allowing significant growth for current product families.

In view of this momentum, it seems appropriate and timely that we dedicate this issue to Micro Channel architecture for the purpose of promoting a better understanding, not only of the architecture itself, but of applications of the architecture with a specific focus on bus masters. In this issue individual authors have the opportunity to describe advantages and functions in their own words.

Our objectives are to:

- Broaden understanding of Micro Channel architecture – including trends and future direction
- Demonstrate its capabilities through applications of the architecture to advanced hardware and software product development

- Demonstrate IBM's intent to increase productivity of independent developers of Micro Channel products through the availability of interface hardware and consistent software protocol
- Demonstrate IBM's intent to support independent developers of Micro Channel products through the availability of detailed documentation design seminars and the IBM Developer Assistance Program
- Recognize technical contributions of IBM and of independent developers

We hope that developers, dealers and users will all find value in the articles in this issue in terms of gaining a better understanding of the power and depth of Micro Channel architecture as a platform for growth today and in the future.

ABOUT THE AUTHOR

Dr. Robert L. Carberry is Vice President of Systems for IBM's Entry Systems Division. He joined IBM in 1964. In the Federal Systems Division he held various management positions, including manager of Systems Architecture and manager of Processor Development. In 1979, he became assistant to the President of that division. Dr. Carberry next joined the Data Systems Division as manager of Processor Development and was subsequently appointed to the position of Director of Engineering and Scientific Processor Programs. In 1982, he was named Director of the Data Systems Division, Kingston Laboratory, responsible for various hardware, operating system, and language development activities. In 1984, Dr. Carberry was named Vice President of Systems Architecture and Technology for the Entry Systems Division. In January 1985, he became Vice President, Product Development and Technology and then in August of that year, he became Vice President of Product Line Management and Operations. In December 1986, he joined the Corporate Development Staff as the IBM Director of Systems Product and Technology. In June 1988, he assumed his present position.

Requirements for Advanced Bus Architecture

Chet Heath
IBM Corporation
Boca Raton, Florida

The structured channel architecture for microsystems has evolved in the same spirit as the structured channel architecture for System/360 mainframe systems. Like the System/360 channel architecture, Micro Channel architecture has become a standard, an accepted architecture for advanced microsystems. Micro Channel architecture is fully compatible with applications and operating systems for the IBM Personal Computer, and it supports a new family of adapter cards as well as new capabilities.

IBM Heritage

The personal computer owes its heritage to a period in 1973 when IBM introduced the 5100 series of products. The 5100 machine was a desktop computer that could execute about 30,000 instructions per second. It had a 5-inch screen and a keyboard that could not be detached. In its time, it was useful for simple, single-task, single-thread applications, typically written in BASIC or APL.

The IBM 5100 machine later advanced to the 5110, the 5120, the 5130, and the System/23 Datamaster. The Datamaster, based on Intel® 8085 processor architecture, differed principally in its channel architecture from the PC by only four lines. The PC featured additional addressing of up to one million bytes.

The PC evolved into the Personal Computer XT™ with the addition of a fixed disk. The Personal Computer AT®, initially operating at 6 MHz, produced a theoretical 750,000 instructions per second. That was later upgraded to one million instructions per second with an 8 MHz AT® implementation. For comparison, the theoretical calculations for several machines are shown in Figure 1.

All these implementations were meant to be compatible with previous products, but they lacked a general architecture or direction. It became obvious with the AT computer that a general structured architecture for advanced microsystems would be needed to move to the multitasking, multiuser environment with concurrency, typified by PS/2 Models 50 and higher.

Previous Evolutions

A similar architecture evolution occurred in 1964 when the 1401 series of machines evolved to the IBM System/360. That transition was from single-task, single-thread operations to multitasking, multiuser op-

erations for which the System/360, its channel architecture, and its entire system design were intended.

Another evolution occurred in 1973 in the minicomputer arena between the IBM System/3, which was a single-task, single-thread environment, and the multitasking System/3X series of machines. A transition was provided at this time – the System/32 was provided to continue the single-task applications in equipment designed for the multitasking environment.

In 1987, IBM accomplished another evolution by making the transition from the single-task, single-thread environment of the AT computer multitasking PS/2 system with Micro Channel architecture, while also providing a compatible environment for single-thread applications in the Models 25, 30, and 30 286 (Figure 2).

Each time, the needs of the future could not be met by modifications of the past, and a new generation of systems was defined.

Approximate Performance Comparison		
	Model	Approximate Performance
Single Tasking	5100	30,000 Instructions/Second
	Personal Computer	300,000 Instructions/Second
	AT (6MHz)	750,000 Instructions/Second
	AT (8MHz)	1 Million Instructions/Second
Multi Tasking	PS/2 - 50/60	1.25 Million Instructions/Second
	PS/2 80	3 Million Instructions/Second
	PS/2 - 70-A21	6 Million Instructions/Second

Figure 1. Approximate Performance Comparison

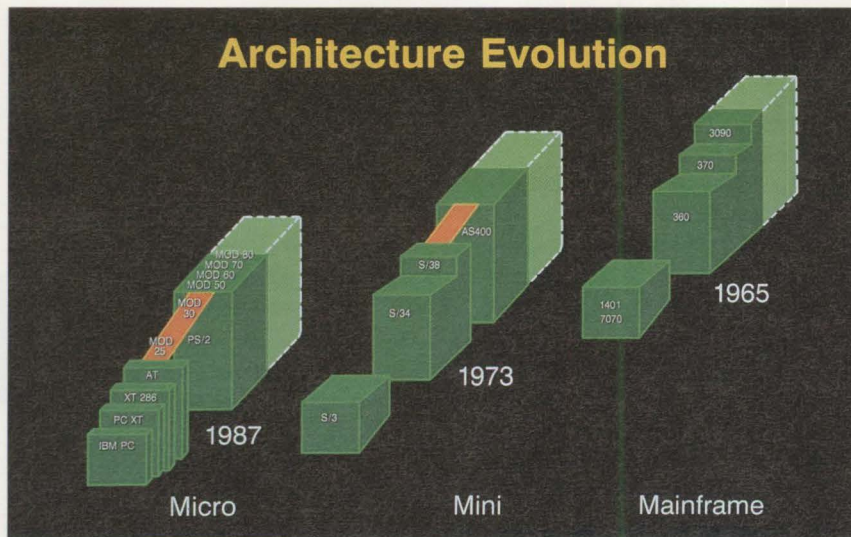


Figure 2. Architecture Evolution

Single-Tasking Versus Multitasking

In a single-task system, each operation occurs in sequence. Much the way a person can assemble a bicycle in sequential steps (first the handlebars, then the wheels, and finally the seat), so can tasks in a computer be manually performed, in sequence, by one user. The time to complete the entire sequence of operations is the sum of the times required for each task.

In the single-tasking environment, operations to the disk require a delay in processing. This is because accessing the disk, talking to the I/O, or operating the processor are each sequential operations. During the seek latency period of the file, the processor is unused.

The command to operate the disk and the time to process the data are typically short compared to the time required for the disk to respond. All other tasks wait for the first to complete, and delays accumulate from all tasks. For this reason, the latency of the disk file in a single-

tasking machine is a significant factor in the performance of the system as a whole.

Single-tasking has the advantage of simplicity in system design, yielding low-cost systems. But it is not the way most of us work or think — most of us multitask in both our work and our thinking.

In a factory, for example, multiple tasks would be performed concurrently to maximize productivity. Operations would be done simultaneously whenever possible. The time to complete the entire operation is then little more than the time required to complete the longest task.

The difference between single-tasking and multitasking involves this concept of concurrency, which means that, in multitasking, I/O operations can occur at the same time rather than sequentially.

When the system is shared among a number of users or a number of tasks, it becomes much more viable economically, and the organization

that it supports becomes more productive.

Technology Trends

Looking at the trend of technological advances from 1981 until today (Figure 3), we see a 5-to-1 improvement in the clock rate of the processor. The efficiency of the execution unit for real-mode instructions has improved by a factor of 4-to-1. We have seen about a 20-to-1 improvement in the processing power of the system. I/O devices that depend on the processor to support data movement have increased their data transfer requirement (called throughput) a great deal more. Displays have moved from 128,000 picture elements to more than 6 million. Direct-access storage devices moved from the 50,000 bytes-per-second transfer rate of diskette media to the enhanced small device interface (ESDI) in excess of 1.25 million bytes per second, and most recently to the small computer system interface (SCSI), transferring data at approximately 5 million bytes-per-second.

Communications has undergone the most dramatic transition in performance. Going from 300-baud modems, which were a luxury in 1981, to 16-million-bit-per-second local area networks has produced a 53,000-to-1 improvement. I/O devices are now moving data much more quickly.

The Need for Change

The objectives for the IBM Personal Computer were to provide an address space of 1 million bytes and a path for the transfer of one character at a time. These machines could do only single transfers of data, which typically depended on the processor for every transfer. The AT was extended to allow a 16-

megabyte address space defined by a 24-bit address bus and a 16-bit data path. The AT computer also introduced the "string mov" operation capability of the 286 processor that allowed the rapid movement of blocks of data to and from file devices.

Single-tasking machines operate their I/O interfaces one at a time, so the requirement for the bus throughput is no greater than the fastest I/O device. But in multitasking, where tasks and I/O devices operate at the same time, throughput requirements are higher.

IBM did not consider the AT bus to be a suitable base upon which to define a family of multitasking and multiuser systems. When taken together, the limitations of the AT's interface to I/O implied a number of concerns about the design of any system that supported compatibility with the cards designed for the PC, PC XT™, and AT computers. In 1984 these limitations may have been viewed as advantages for the single-task, single-thread environment; indeed, the AT is still a qualified bus for those operations. But in the multitasking environment, there were limitations in several areas:

Addressing – 24 bits: The AT implemented only 24 address lines for memory, so only 16 MB of storage can be addressed on the bus.

Data Bus – 16 bits: The AT data bus is only 16 bits wide, limiting the throughput, when existing cards are installed, to 5.3 MB instantaneously and about 3.5 MB continuously.

Connector Design and Insertion Force: The PC and AT connector design is a legacy from the mid-

1970s' design of the IBM 5140, with insertion and removal forces approaching 40 pounds. Good design practice would keep the insertion and removal forces below 40 pounds for assembly and service operations by humans or robots. Extending the AT connector to have sufficient pins for a 32-bit interface would yield forces closer to 65 pounds during either insertion or removal. This would expose the system board to excessive stress during assembly and service operations.

Direct Memory Access (DMA)

Utilization: The DMA channels, used as alternate paths between I/O and memory, were permanently assigned to devices early in the development of PC and PC XT machines. The AT design added more channels at a wider width, but DOS cannot use them. As a result, the DMA controller is assigned to

I/O that is slow by today's standards. The processor does all the work to move data between memory and I/O.

Electromagnetic Compatibility

(EMC): The radiation of electromagnetic energy outside the system and susceptibility to external interference are not just problems to be solved by the manufacturer, but issues of data quality and validity for the user. When energy is radiated from a data signal, the data levels are reduced, and may oscillate below valid levels. The signals then become vulnerable to outside interference. Shielding the cabinet is expensive, and only contains the energy within the cabinet. It does not limit the electromagnetic incompatibilities between cards or devices within the cabinet. EMC problems increase, as the square of the data rates increases, on the I/O bus.

	1981	Today	Ratio*
Processors	8088	80386	4:1
Instruction Unit Efficiency			
Clock MHz	4.8	25	5:1
Maximum Memory Size	2 ²⁰	2 ³²	4000:1
Data Bus	8	32	4:1
Performance	300 KIPS	6 MIPS	20:1
Display (Mbits/screen)	0.128	6.291	50:1
DASD (MBytes/sec transfer rate)	0.050	1.250	25:1
Printer (Bytes/sec)	80	5000	63:1
Communications (Bits/sec)	300	16,000,000	53,000:1

* Approximate performance ratios for comparison only

Figure 3. Technology Trends

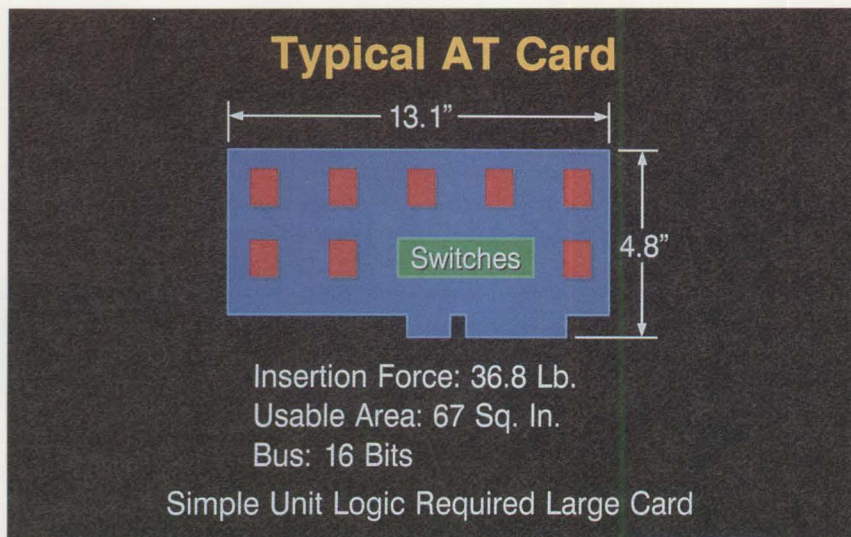


Figure 4. Typical AT Card

Faster I/O demands a solution to this problem.

Configuration Conflicts: Because the system resources required to operate a card are either fixed in the design, or very limited in selectability by switches, and because most PC, PC XT and AT designs decode only 10 of the 16 I/O address lines, many conflicting situations exist with the configuration of systems that use these cards. For example, because PC, XT, and AT interrupts cannot be shared between two device adapters, and because compatible software typically recognizes only two communications interrupts, a maximum of two cards that drive the interrupt request lines can be installed. If a user wishes to install a binary synchronous adapter, one asynchronous card must be removed. If a synchronous data link control (SDLC) adapter is installed, the binary synchronous card must be removed. This limits the number of solutions that the user can define.

Logic-ground Isolation: Personal computer cards have only three ground pins to connect the logic

ground on the card to the system board ground. If a card pulls sufficient current from the system, the logic-ground reference on the card can rise as high as one-half volt above the system ground. Data transfer between the cards and the system can then yield misinterpretations in data level, and invalid transfers can occur. Therefore, if a branch address is sought at the instant that a noise spike or static discharge pulse from elsewhere in a network is added to the DC offset, and if the branch is taken, the system can lose its place in memory and execute variables or instructions out of order. The user is locked out of the keyboard. This places a requirement on any new architecture to solve this problem.

Processor Workload: Affecting performance the most, however, is the loading of a great deal of the I/O-to-memory transfer burden onto the processor by simple PC adapter cards that implement little hardware and use the processor to perform much of the interface logic through the basic input/output system (BIOS) or application drivers. This

practice may be permissible in a single-tasking machine because the processor is not used elsewhere in a single-tasking system. But in a multitasking machine, such I/O designs can decrease performance because they tie up the processor when it could be dedicated to tasks and the operating system.

The limitations that compatibility to PC, XT, and AT cards would impose on a system outweigh the relatively infrequent need to propagate old cards from an old system to a new system.

Getting Technical

Mechanical Considerations:

Taking a closer look at the system design limitations just outlined, consider the typical AT card (Figure 4). When populated with early 1980s-vintage unit logic components and switches, a large card was required to perform even the simplest I/O adapter function. The personal computer 8-bit card connector consumed approximately 3 inches, or one-fourth the width of the PC system board. The AT extended the connector to slightly less than half the width of the AT system board, and tightened the mechanical reference and tolerance control in the placement of the copper tabs on the cards and in the spring contacts in the system unit connector. The force to insert or remove the card grew close to the 40-pound maximum recommended force. Large connectors consumed a large fraction of the limited system board area, yielding a large system board when the processor, memory and I/O adapter logic were added to the layout (Figure 5). This caused the system unit to consume more area and a larger fraction of the already crowded desk. And, from a manufacturing point of view, the larger

AT cards may yield unusable waste when they are cut from the standard 12-inch by 24-inch raw stock.

It became obvious that a smaller system overall, with smaller components that consumed less desk space, would be an advantage to the customer.

While this requirement may imply more development expense and time for adapter card designers to implement in a smaller space, that expense is typically amortized over tens of thousands of cards. Indeed, the techniques typically employed to implement in a small space – large-scale integration, surface-mount technology, automated manufacturing systems and computer-aided design – yield card designs that are often less expensive to duplicate and more reliable.

The developer's challenge is not the customer's problem. To the customer, small cards are better, if a smaller footprint is needed.

Electromagnetic Compatibility:

Let's unfold a card to visualize the EMC design problems of AT cards and the systems that they support. Looking specifically at both sides of the connector interface at once (Figure 6), and highlighting the radio frequency ground signals, one can see that most of the signals do not have the close proximity of a radio frequency shield. In fact, the processor clock was placed midway between two radio frequency grounds. This was acceptable when the card format was designed in 1978 for the 5140 machine, from which the PC was derived. That system had a slow clock by today's standards.

Today, however, clock rates are much faster, data transfers are much faster, and I/O devices move much

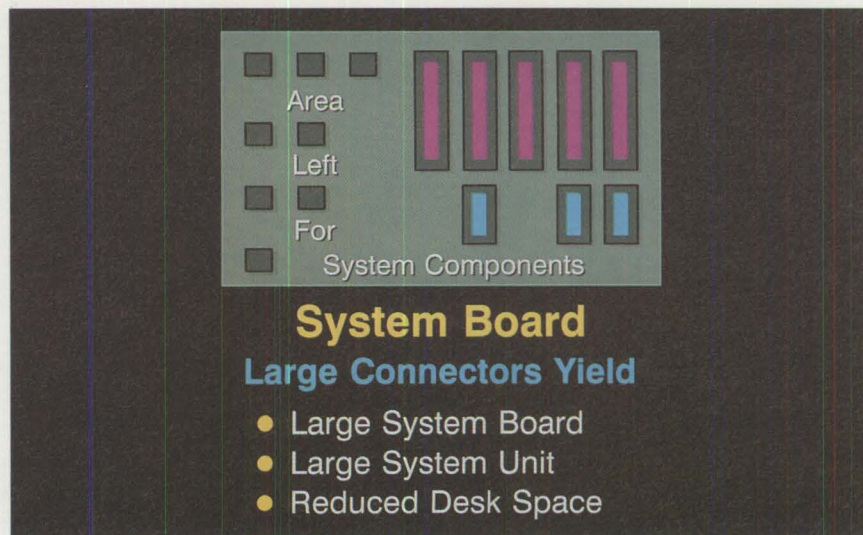


Figure 5. System Board Layout

more data in a shorter time. The result is electromagnetic incompatibility with other nearby equipment and internal interference between elements within the system cabinet. The security of the data moving through the machine can even be compromised by others.

To prolong the life of the PC bus and allow extension to the 16-bit AT interface, IBM introduced "sin-

gle-point ground" power systems, conductive plating of the power supply, frames and covers, and internal ground planes in PC XT and later machines. But those measures become much more costly and less effective as the speed of the systems increase. To understand why, let's go back to basic electricity.

Many of us, in our youth, built electromagnets by winding turns of wire

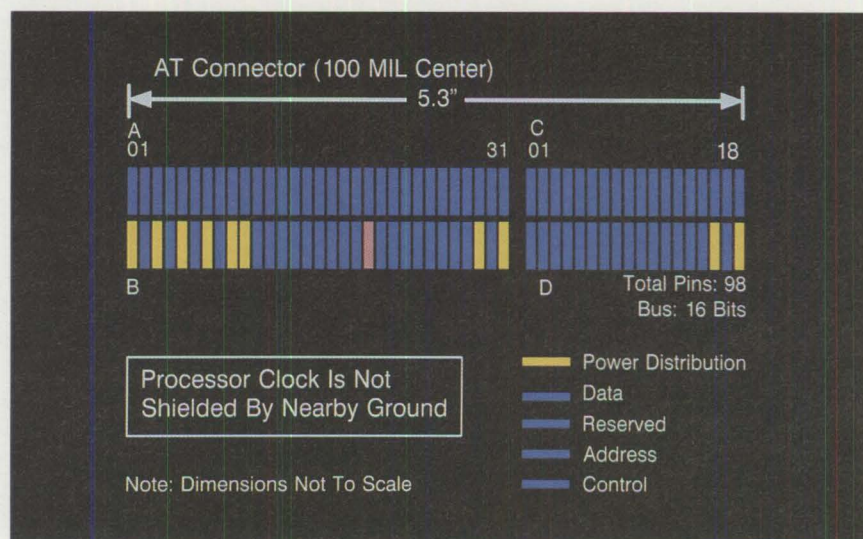


Figure 6. AT Connector

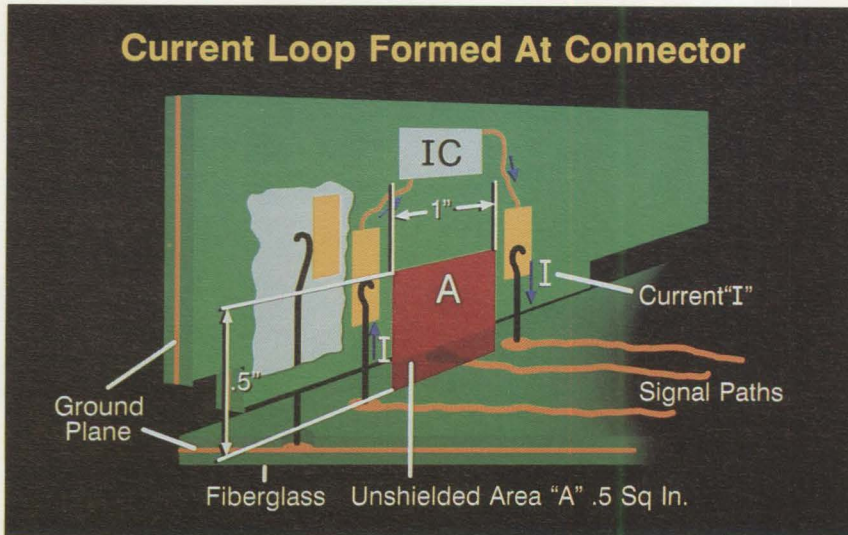


Figure 7. Current Loop Formed at Connector

around an iron nail or spike. We found that as we increased the area inside of loops by creating more turns or accumulating area, the magnetism increased. Also, when we increased the current by adding more batteries, we increased the magnetism.

This is true for a fixed distance from the radiating element. The electrostatic portion of the radio wave is proportional to the area inside of a loop times the current that flows through that loop, and if it is an alternating current or radio frequency, times the square of that frequency.

The data transitions on the computer bus are currents that alternate at very high frequencies and thus have the potential to transmit energy (or receive energy) as interference with other equipment either inside or outside the box.

Look at the card connector in three dimensions (Figure 7). An unshielded area is formed where the current flows on spring tabs to a card, through the card, and then re-

turns over spring tabs back to the system. Everywhere else in the system – on the cards and on the system board – there is ground shielding, except at this one point.

Using a radio frequency quiet room, IBM engineers found that the card connector on the system board was the principal source of radiation from the I/O bus when the interface between I/O and the system was active. This unshielded area was considerably larger in the PC, PC XT and AT implementations – about one-half square inch. In all these implementations, the ground system on the cards is connected to the ground system or shield on the outside of the case.

However, this meant that when the limited number of grounds in the AT was inadequate to sink a current back to the power supply – for example, if the card bracket retaining screw was not tightened firmly – the currents could flow back through the metal system frame (Figure 8), which is the intended path for containing noise in the PC, PC XT, or AT system. Current re-

turning to the power supply could actually flow out over the shields of cables and back through the AC power mains, forming a gigantic loop called a "ground loop." Even with a small current flowing, this would create a large loop area and very large radio signal. This is called "conducted electromagnetic interference."

The ground mechanism used in the PC, PC XT, and AT also makes the system susceptible to noise from outside the system. Energy from equipment connected to the system could be coupled through the shield system. If the retaining screw was not tightened, the current could sink through the card and through the system rather than safely returning to the AC mains. Typically, this would manifest itself as a data error when static discharges, perhaps from someone touching adjacent equipment in a low-humidity environment. This data error would not necessarily occur in the equipment that the person touched. In extreme cases, a lightning strike could actually destroy local area network (LAN) cards or communications cards where the screw is not tightened.

Requests for Processor Attention:

The AT system unit typically had a few devices connected to it, enough for one user and a few applications. It was designed to support simple file serving and very limited multi-tasking configurations for only one user. Yet, increasingly, we are using desktop systems as small mainframes. With multiple users often attached through networks, and multiple tasks per user, these systems often perform the same duties as the mainframe machine in a raised floor environment.

In a mainframe environment, each user or each task may have a favorite piece of I/O that it operates. For example, if two tasks were to talk to the same printer at the same time, the text would be mixed between the two, and we would see gibberish printed on the paper. Obviously, BIOS or an operating system would prohibit this error by sequentially scheduling access to the printer – but this would result in one task delaying another. Still another task might require a special printer or plotter for output. This is one of the reasons that the total amount of I/O and types of I/O attached to a system can grow when multitasking and multiuser operating systems are introduced.

To receive requests for attention for I/O devices, the AT has 11 interrupt request lines. These signals on the AT card interface cannot be shared between device adapters. This means only 11 different interrupting devices can be configured in an AT system.

In a multitasking system such as the Personal System/2 Model 80, where there are eight connectors and six adapters on every system board, we are talking about 14 functions. Here, eleven interrupts would be insufficient. A redefinition in the AT interface with at least 14 interrupts would be required to solve this problem alone.

Why can't interrupt requests from PC or AT cards be shared? There are two reasons. First, let's understand how the edge-triggered interrupt request mechanism works in systems that support personal computer cards. A transition from low to high on a PC bus interrupt request line is interpreted by the system as a request for attention by an I/O adapter. To create this transi-

tion, the request line is first driven low, then high. The drivers on adapter cards may be inactive at other times, but they, too must drive low, then high to get the transition.

A driver with a pull-down transistor and a pull-up transistor is required to drive the interrupt request line quickly from one state to the other. A bipolar or a tri-state type of driver may be used, as long as the high state immediately follows the low state. First the bottom transistor becomes active, "on" like a switch. Then it turns off as the top transistor becomes active, causing the transition.

Assume two such drivers on separate adapter cards are connected to the same interrupt request line. Eventually one adapter will make a transition. While the first adapter is still driving the line high, a second adapter drives the line low as it begins to present an interrupt. This type of situation is much more likely when multiple I/O adapters are operated simultaneously by concurrent tasks in a multitasking or multiuser situation.

During the overlap of the two simultaneous requests, a temporary short circuit across the power supply is created. This situation would also occur if one of the drivers were an open collector driver, as used in level-sensitive interrupt systems. This may cause either one or both of two problems:

- 1) The interrupt-request line may not achieve a valid level as the two transistors fight to pull it both low and high at the same time. The interrupt may thus be lost to the system. A lost interrupt can be a catastrophe in a multitasking operating system and may require a reset and restart of the system hardware, to regain synchronism between the I/O and the system.
- 2) With time, perhaps milliseconds, or perhaps much longer, the drivers may be damaged and the lifetime of the components would be shortened, requiring the replacement of a card or a system board that contains the driver.

A resistor between the card and the bus could limit the current, but

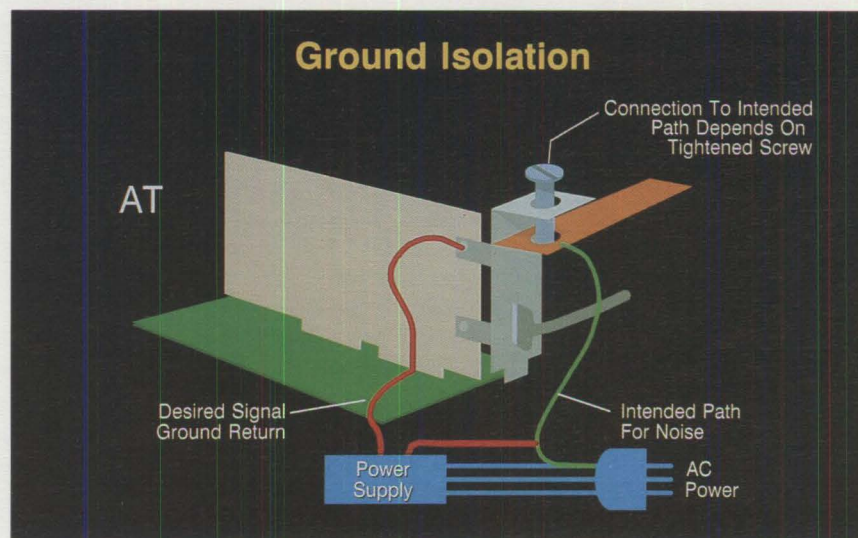


Figure 8. Ground Isolation

would then decrease the noise margin for the logic and aggravate the noise problems in the system.

Schemes have been devised to inhibit one card driver, if an edge is detected from another interrupt driver on the same request line. They may work when two interrupts are unlikely at the same instant in time, but they are also sensitive to noise spikes on the interrupt-request lines. The noise spikes may be detected by many or all such circuits, inhibiting some or all adapters. No interrupt is present, none will be serviced, and the system can hang.

In the opinion of IBM engineers, mixing the drivers on existing PC, PC XT, and AT cards, or mixing existing interrupt drivers with new cards that employ only open-collector drivers, could be detrimental to both data and system integrity, as well as to the reliability of the system. Consequently, we concluded that any system that mixed level-sensitive and edge-triggered interrupts on the same request line may be unreliable for users and very difficult and potentially expensive to support under warranty and service agreements. Even if all the possible warnings are given, normal human errors would still result in unreliable configurations.

A satisfactory design point could be achieved only if all existing PC, PC XT, and AT cards were excluded from an interrupt-sharing mechanism and were defined with an all open-collector, level-sensitive interrupt sharing system. *One of the greatest advantages of such a system would be that it does not support existing PC, PC XT, and AT cards.* Given that all new cards would have to be designed, and a new system would have to be designed to support them, the best

technical solution could be chosen without compromise for PC, PC XT, and AT card support.

Requirements of the Multitasking Environment

Many other limitations imposed by support of PC, PC XT, and AT card designs could be fixed as well.

The Single-Tasking Environment:

One limitation is the excessive dependency of most I/O devices on the processor for data movement and control. The displays, all the files, most communications, most local area networks, and printers are

*The best
technical solution
could be chosen
without compromise.*

dependent upon the AT processor to move data. An 80386™ processor, with all of its complexity and sophistication, is reduced, for many of the operations in the machine, to simply moving data and operating I/O devices. In a single-tasking environment, it may be acceptable to operate each I/O device in sequence and to dedicate the processor to support that device. Such a system may even look very favorable in single-tasking benchmarks. The I/O adapters can be relatively simple and inexpensive to build.

The Multitasking Environment:

However, in a multitasking environment, all of the interrupt-driven I/O adapters tax the system performance before the applications and operating system instructions can be exe-

cuted. Before PS/2 systems with Micro Channel architecture were developed, the solution to supporting fast, processor-dependent I/O was typically to build faster processors and faster memory systems – a very cost-ineffective solution.

For example, let's look at what happened with printers. In 1981, an 80-character-per-second printer was considered cost-effective and adequate for a personal computer environment. Today, we have moved to laser printers that can transfer 5,000 bytes per second in graphics mode for desktop publishing. In a DOS environment, this original load of 80 bytes per second on the processor was rather modest. DOS applications consume about 200 instructions to service the interrupt, and to move a byte to the printer. These instructions save the state of the machine, do the housekeeping required to figure where the next print character will come from, operate the interface, and resume processing at the place that it was prior to the interrupt. This requires one interrupt per character in the existing printer adapter. The asynchronous and binary communications adapters, and many other adapters for the personal computer, are of this same basic design, with one character transfer per interrupt. This means a load, under DOS, of 16,000 instructions per second. This is a modest load, because there are 300,000 instructions per second available in the PC.

The same function today for a single interrupt is a load of 5,000 characters per second, times 200 instructions per character, or 1 million instructions per second – about all of the capability of a 80286™ processor, running at 10 MHz.

Under a multitasking operating system, however, the environment is much much larger and more complex; it can take 1,000 instructions to service the interrupt. This means a 5,000 character-per-second printer will use 5 million instructions per second. That is approximately all of the capability of a 25 MHz, zero-wait-state 80386-based system. So a \$10,000 computer is tied up 100 percent of the time when the printer is operating.

Processor Bottleneck: A requirement of concurrency is that multiple devices can operate at the same time. If each device depends 100 percent on the processor, then two devices cannot operate at the same time at full performance. One solution to the burden on the processor is to offload that burden to a direct memory access (DMA) controller. Then multiple channels of the DMA controller can each simultaneously support an I/O adapter while the tasks and operating system get maximum access to the processor. This would be the ideal solution. Unfortunately, it is not possible with the AT bus and may be severely limited in a system that supports AT cards.

When the personal computer evolved in 1981 from the original mid-1970s Datamaster product, it implemented a DMA control function. In 1981, that was considered an advantage over competitive designs. The diskette moving data at 50,000 transfers per second was the fastest I/O device. It was assigned to DMA channel 2.

Cards using the DMA controller used the same drivers as interrupt request drivers and consequently could not share a DMA request line.

The connection to the DMA request and acknowledge signals on the PC

bus was also hardwired on the card, to assigned DMA channels. The result was that each new DMA adapter design would require that a new DMA channel be defined. Otherwise, an installer would need to understand which card designs, which DMA channels, and which configurations to avoid, to protect the drivers.

The SDLC adapter was similarly assigned to DMA channel 1. Dummy-read transfers from memory using DMA channel 0 did the refresh operations. This left only DMA channel 3 available for further implementations. When the PC XT was intro-

Multiple devices can operate at the same time.

duced, DMA channel 3 became the interface for the fixed disk. The PC Network was defined to allow the cost-effective sharing of fixed-disk storage.

It may seem as though a rule has been violated here, where the PC Network and fixed disk share a DMA request line. To avoid problems, the PC Network BIOS turned off the fixed disk when the network was active, preventing two drivers from simultaneous activity on the same line. The DMA request was sequentially shared, and file and network activity became mutually exclusive. This was permissible. Remember that the PC XT was a single-tasking design, like its prede-

cessors, and did not support application concurrency.

When the AT computer was introduced, it could offer concurrency between the fixed disk and PC Network by moving the fixed disk adapter to the processor and using the "string mov" capability of the 80286 processor. The display controller, asynchronous communications, printers, and bisynchronous communications were getting faster and were an increasing burden, but the overhead to support them was still within the capability of the faster 80286 processor.

Networks evolved connections to other networks called "gateways" or "bridges." This implied two network functions in one system. The mechanism of having the PC Network BIOS turn off another device would not work when the other device was the PC Network itself. Either more DMA channels would have to be defined, or the network responsibility would have to be moved to the processor, where the attention could be moved between I/O adapters with interrupts, albeit with considerable software overhead.

The AT computer did both. It defined three new 16-bit DMA channels and moved networks to the processor along with second and third fixed-disk devices, using the power of the processor to support I/O. Yet the processor was now the single element that could move data for most I/O. As long as faster processors and memory systems were cost-effective, the operating system could schedule concurrent tasks as sequential tasks, and the speed of the processor would make up for the fact that all I/O would need to wait in line for the one processor (Figure 9). The AT also removed the memory refresh from DMA

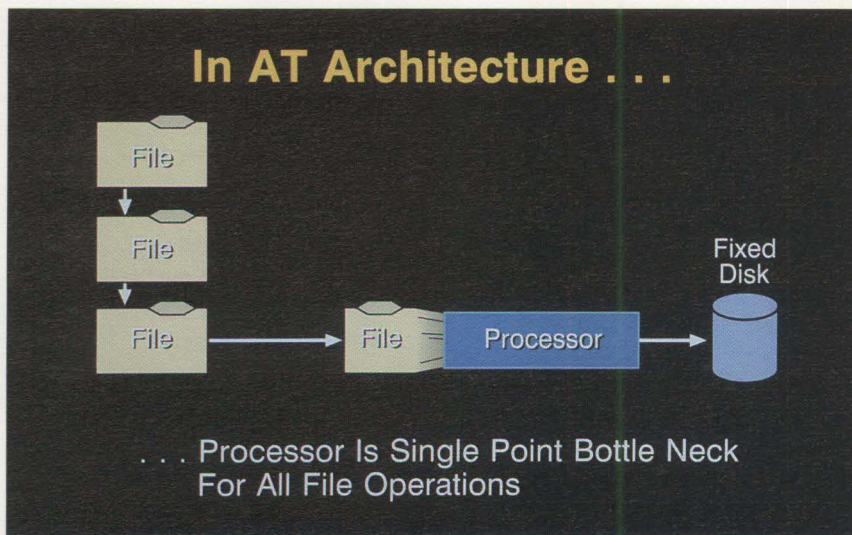


Figure 9. In AT Architecture

channel 0. This left four DMA channels free for use by I/O devices. But for the large part, none of the new channels has been used. Why?

Originally intended to support only sector-oriented files, the new 16-bit channels were incompatible with DOS. The reason is that the DOS file system is byte oriented, and it can place a buffer of data anywhere on an 8-bit boundary in storage. It can create buffers that are even or odd, and they can end or start on even or odd boundaries.

The Intel 8237 DMA module was implemented to count words, not bytes, on the AT and would only allow DMA transfers from even boundaries. This is a characteristic problem of many systems, derived from the AT architecture, that implement the 16-bit DMA ports.

DMA channels 5, 6, and 7, if implemented in those systems, are incompatible with DOS because they cannot place a buffer on any given boundary as defined by the operating system. DMA channel 0, which

was freed up in the AT because refresh responsibilities went to dedicated logic, has not been used. Perhaps this is because card designs would have to be specific to the AT, and the AT only, and could not operate on personal computer designs.

To summarize these points, the AT system has three active, or usable, DMA ports for which there exists a business case to build cards that can also be used in the PC and PC XT implementations. This limitation on the ability to move data transfer responsibility from the processor is a major limiting factor in the concurrent environment of a multitasking operating system such as IBM Operating System/2™ (OS/2).

All Things Change

The Complete System: In a concurrent environment, typically a number of applications are operating at the same time. This is the intent of a multitasking operating system – to do more than one thing at a time. It is what can make the Personal System/2 more productive,

as much as five times more productive, than previous machines.

We have accepted that a new operating system is required to run these multiple applications concurrently. We have accepted that new processors are required to do multitasking and multiuser operations. The 80286, 80386 and 80486™ processors allow the efficient switching of tasks because they have hardware dedicated for those functions.

Multiple tasks create simultaneous activity to a number of I/O devices. Each application may operate a unique device; the number of devices increases. More types of devices are connected, and the amount of data traffic, or throughput, to and from these devices expands. The number of requests per second for processor attention, interrupts from these devices, grows as well.

The path between I/O and memory, the channel, must be redesigned to reflect the change. Every element of the system – adapters, processor, operating system, applications and the channel – must be redesigned.

These characteristics of concurrent systems are not new. We have seen them in every multitasking and multiuser system that IBM has produced since 1964 (Figure 10).

OS/2 is not the first operating system that we have used for multitasking. In fact, multitasking is at least 25 years old. The problems that micro system designers are finding today are often similar to problems that minicomputer designers saw 15 years ago and mainframe designers saw a decade before that. What *is* new is that the rules have changed and innovative concepts have been applied. Logic can now be more complex; it is faster, less expensive,

and far more reliable. The processors are more powerful. Memory is far less expensive. We can be creative and use our experience as well.

Video Graphics: The reason that the change can occur now is technology. Figure 11 shows a comparison between an Enhanced Graphics Adapter (EGA) card that was designed around 1984, and a Video Graphics Array (VGA) chip that replaces that function. In fact, the chip package consumes almost 80 percent of the chip's area; only 20 percent of that component is the actual VGA silicon chip. The VGA chip has almost twice the function of the EGA card.

The display function is one of the most frequently upgraded features in a system. Unfortunately, another limitation of systems designed for AT cards is that the investment in the display adapter will probably be discarded within a few years. Why?

When the EGA card was installed, the monochrome and color cards were removed. The original investment was lost as the new EGA function made the old cards obsolete. The user paid again for the monochrome and color functions because half of the EGA logic was dedicated to reproducing the old display modes. AT users upgraded to VGA and paid again for the EGA display function. Each time, approximately one-half of the investment in the new card was to replace the function just removed. Micro Channel architecture puts an end to this loss of investment, as we will see later.

File Servers: A file server is a potential single point of failure for what is, in effect, a multiuser system. It operates I/O concurrently, and is an example of the type of sys-

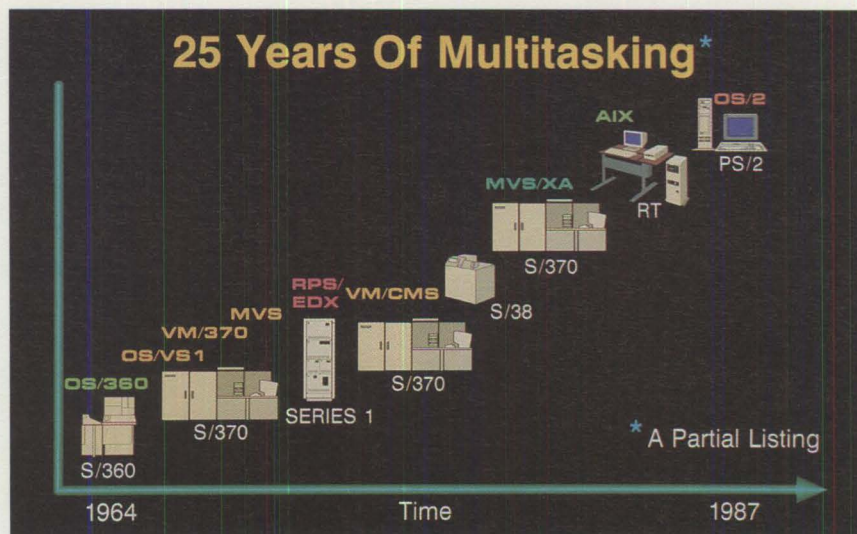


Figure 10. 25 Years of Multitasking

tem that needs Micro Channel architecture today.

The server will probably grow in its throughput and configuration as more I/O devices and users are connected to the LAN. The desktop will use Micro Channel architecture to run multiple tasks at the user station and to allow network asset management or control of the network configuration. Network asset man-

agement allows an electronic census of installed options, prior to distribution of applications, that may be I/O device-dependent. Additionally, the network manager can cycle adapter functions in the LAN stations on and off to accommodate application I/O needs and remote diagnostics. These functions can be supported by specialized software applications in all the stations and the network

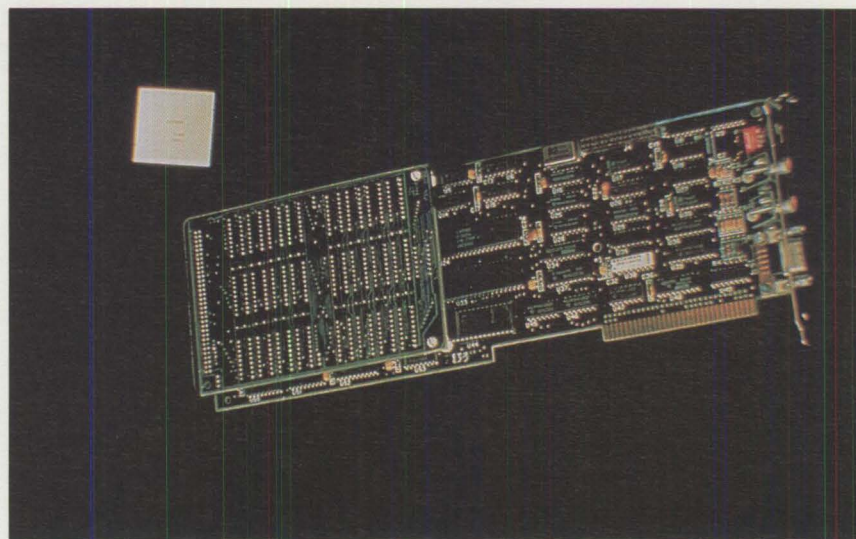


Figure 11. Video Graphics Technology

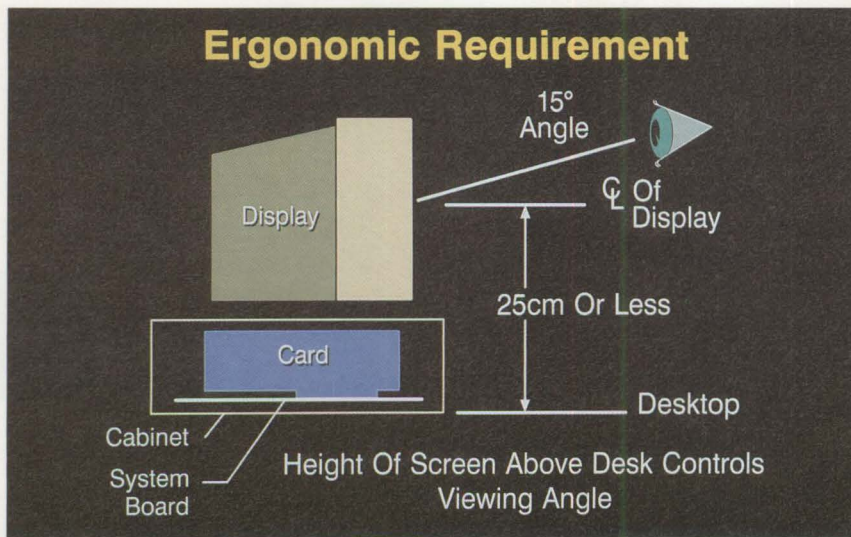


Figure 12. Ergonomic Requirement

manager or by the operating system itself.

A New Foundation

An architecture for the 1990s has 32 bits for both I/O and memory, can be expanded to wider data bus widths, allows higher throughput, and still supports the 80286 and 80386SX™ and other processors with 16-bit interfaces. It is independent of both the processor and operating system. Specifically, IBM's intention was to create a foundation for a broad range of machines, with an architecture that could support single or multiple bus system architectures (for example, the PS/2 Model 70-A21).

EMC and Signal Quality Characteristics at Higher Speeds:

The channel would be expandable to wider data paths and faster throughput. It would allow advanced error detection, recovery and diagnostics.

To support the faster data transfer, the data quality must be predictable

and not lose integrity through radiation.

Automatic Configurability:

Complex multitasking and multiuser systems would need automatic setup to support conflict-free configuration of multiple I/O interfaces. If IBM had kept switches, resolution of conflicts in configuration would have been very complex indeed. Assume that a system had eight sockets; then one could install as many as eight cards of the same design. That would mean eight I/O address ranges for the adapter control registers. Three switches are needed to select one of eight options.

To provide enough DMA ports to move processor-dependent I/O off the processor, four switches would be required to select one of 15 DMA ports. There are 16 address locations, for "on-card ROM," used for power-on self-test (POST) and BIOS extension – another four switches. Without interrupt sharing, three switches would duplicate the seven "PC" interrupts, but four would be needed to select one of all

11 non-shared AT interrupts. This is a total of 15 switches for each adapter:

3 I/O addresses
4 DMA channels
4 On-card ROM/RAM addresses
3 or 4 1 of 11 interrupts

15 Switches

Designing for the worst case, with six adapters defined on the system board and four adapters per combo card in each socket, as many as 570 switches would be required for fool-proof, switchless set-up on the AT computer:

(6 x 15) System board
+
4 x 15 Each card (combo)
x 8 Cards

570 Switches

This is beyond the average person's clerical ability and patience, but well within the capability of the computer. Programmable Option Select (POS) can replace 32 equivalent switches (expandable to 256,000 switches) and turns what would have been an ordeal for a human into a minor task for the computer.

Smaller Footprint for Desktop:

The smaller card can yield a smaller system, and also can assist in the usability of the system. In several countries, industry regulations and public laws define certain ergonomic characteristics of data entry terminals and personal systems. One part of these requirements is the placement of the centerline of the display no more than 25 centimeters above the desktop to reduce glare and neck-muscle fatigue (Figure 12). With the common practice of pancake-like stacking of the display

and system unit to conserve desk space, the designer has a choice of a smaller screen or a smaller card.

Micro Channel cards are 3.5 inches high, the same height as the connectors to peripherals on the rear bracket of PC, PC XT, and AT cards. PS/2 systems, therefore, maintain connection to most AT peripherals while permitting larger-size displays to be installed where ergonomics is a consideration.

However, Micro Channel architecture does not restrict the card height to 3.5 inches. While this is a specification for existing PS/2 computers, systems with capability for larger cards are feasible.

Increased Flexibility for Interrupt Structure: The AT interrupt structure had to be improved. Micro Channel systems support up to 256 adapters to share a common interrupt-request line. With 11 compatible interrupt-request lines, this allows more than 2,800 device adapters to be installed in a system:

$$256 \times 11 = 2816 \text{ Interrupting adapters}$$

Certainly this is enough for an advanced microsystem; it is also enough for minicomputers and mainframes as well.

The I/O device's request for attention is interlocked with the system's response to the request. Lost or spurious interrupts can be detected, and diagnostics can locate defective adapters and/or interrupt routines more easily.

Increased Data Rates: The Micro Channel interface supports a base burst speed, in its initial definition, of 20 million byte transfers per sec-

ond. Even other microcomputer manufacturers specify the AT bus at no greater than 3.5 million byte transfers per second. There have been implementations of the Micro Channel standard that move data at a rate of approximately 17 million byte transfers per second, or on the order of four times the capability of the AT bus.

IBM has defined all the timing, loading, capacitance, and electro-magnetic characteristics of the system so that designers building cards, systems, and applications to the Micro Channel specification can define equipment that can be integrated properly.

The performance of the Micro Channel bus can be extended well beyond its present level. We have preserved the ability to extend Micro Channel functionality, in a compatible fashion, so that cards developed today to Micro Channel specifications can run on Micro Channel systems tomorrow.

Expanded DMA and Bus Masters for Concurrency: We have expanded the DMA and bus master capability so that a number of devices can be intelligent and no longer be dependent upon the processor. Up to 15 devices can be supported independently of the processor.

Building In the Future

To see how this has been done, look at a physical comparison of an AT card and a Micro Channel card (Figure 13). On initial inspection, the Micro Channel card is physically smaller. This allows for a smaller system. However, the card format is optimized for surface-mount technology (SMT). The connector dimensions match the tab spacing on SMT components, facilitating wiring of the card.

SMT allows an increase in the vertical and horizontal densities, each by a factor of 2-to-1, for an overall aerial density improvement of 4-to-1. Surface mounting also allows the implementation of components on both sides of the board for a total

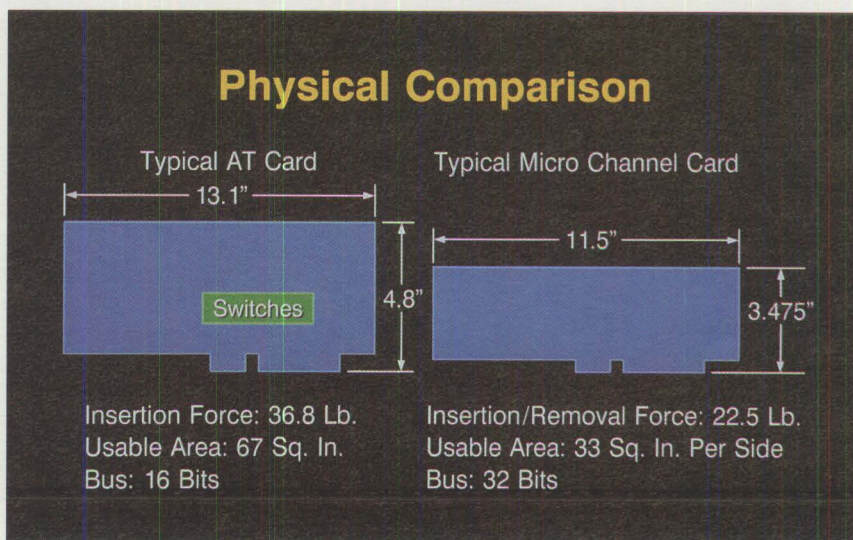
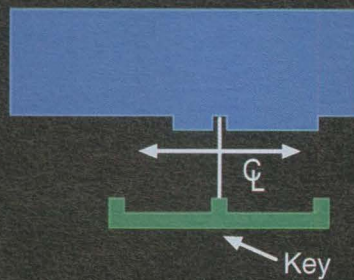


Figure 13. Physical Comparison

Micro Channel Connector



- Mechanical Reference In Center
- Female Can Be Larger Than Male
- 64, 128, 256 Data Path Possible
- 32 Bit Can Be Designed To Insert Into 16 Bit

Figure 14. Micro Channel Connector

improvement of 8-to-1 in component density.

The Micro Channel connector uniquely references mechanical alignment from a keyway centered in the connector, allowing tighter control of mechanical tolerances (Figure 14). When inserted in a 32-bit socket, neither end of a 16-bit male connector is in contact with the end of the female socket. Likewise, a 16- or 32-bit card could be inserted into a 64-bit or larger socket and align perfectly. A 32-bit card has been designed that leaves clearance at the end bulkhead position for a 16-bit connector. Therefore, this 32-bit design can be used in 16- or 32-bit or larger systems. A single design can be made for all systems and still be capable of 32-bit throughput.

The spacings on the connector have been cut to one-half of the spacings used on the AT connector. A Micro Channel 16-bit socket is approximately the same size as a PC 8-bit socket. The Micro Channel 32-bit connector is actually smaller than the AT 16-bit connector. This fol-

lows the trend of other components in the system, such as processors that have grown in power from the 8-bit interface of the 8088 processor to the 16-bit 80286 and the 32-bit 80386 without growing significantly in physical size.

The number of signals in the connector has been doubled, and the insertion and removal forces have been cut almost in half. This is extremely important because in manufacturing and service situations, had the length of the connector increased or more tabs been defined, either the insertion force or removal force would have exceeded 40 pounds.

We have designed out the electromagnetic compatibility problems of the AT connector. One can see by the pattern in the lower half of Figure 15 that each signal in the Micro Channel bus is situated between a ground, across from a ground, or adjacent to a ground signal. There is a radio frequency shield on every fourth pin. On the opposite side of the connector, the pattern is identical but offset by two pin spacings.

The area between the closest shield and a signal is thereby diminished significantly.

The loop area of the Micro Channel bus is one-twenty-eighth of that for the AT bus. The signal transitions can then increase in frequency, by a factor of the square root of 28, or approximately five times (Figure 16). This has been experimentally verified outside of IBM – studies of the Micro Channel bus have shown a cutoff frequency above 80 MHz. Studies of the AT bus have shown a cut-off frequency of approximately 16 MHz, or again the same factor of 5-to-1.

Micro Channel cards now make positive ground contact with an intended path for ground return in the AC power lines. This occurs when fingers attached to the shield bracket on the back of the connector make direct contact on insertion.

The card itself is isolated electrically, so that currents on the card cannot escape that card through the shield system and must return directly to the ground of the power supply. External static discharge currents will be returned through the intended path to the AC power system, not through the card itself.

More DMA: Remember the problems with DMA utilization on AT cards? Only one channel on the AT was usable. The DMA channels that were added to the AT were not compatible with DOS. Micro Channel architecture has solved these problems. Instead of three usable DMA channels supporting just three permanently assigned devices, we can support up to 15 devices with no fixed assignments.

All channels are compatible with DOS. Micro Channel DMA control-

lers can handle the even transfers (as does the AT) as well as all data formats involving odd transfers (those beginning or ending on even and odd boundaries). The DMA controller automatically adjusts the address boundaries upon transfer.

Processor-dependent I/O can now move to DMA because usable DMA channels are available; no assignments are fixed. Data can move faster. Single-transfer operations are employed for compatibility with the AT operations, but we also defined a burst capability in the Micro Channel interface that allows 312 transfers to occur with the same arbitration overhead that normally would have allowed only one data transfer in the AT.

More Designs: Existing designs for the AT system can be migrated to the Micro Channel standard because all of the AT signals can be derived from the Micro Channel interface. However, new functions not possible in the AT can also be defined.

Potentially, more Micro Channel adapter designs are possible. Recall that, on the majority of the cards designed for PC, PC XT, and AT systems, only 10 of the 16 I/O address lines defined by the processor are decoded. This means that only 1,024 registers can be selected by the processor instead of the 65,536 defined by the full 16-bit I/O address.

Even the most rudimentary adapter will define three registers – one for commands to the attached device, another to transmit data to or from the device, and one last register to read status. If these registers are only 8 bits wide, then a maximum of 341 unique designs could be defined for a system that could install

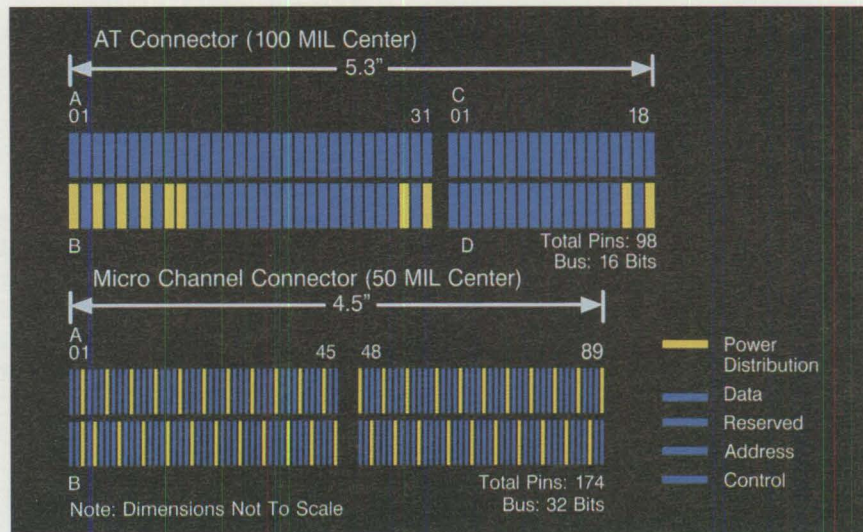


Figure 15. Card Connectors

PC, PC XT, and AT cards with 10-bit decodes.

However, all cards complying with the Micro Channel specification fully decode the 16-bit address, so that theoretically about 22,000 unique designs are possible for Micro Channel systems.

Faster DMA: Micro Channel architecture also defines a serial DMA

capability, much like the "string mov" operation in some Intel processors. Separate read and write operations allow memory-to-memory operations, I/O caching, and inline processing to be defined in the DMA controller. Yet, on first inspection, serial DMA might appear to be slower than the parallel DMA, defined for PC, PC XT, and AT im-

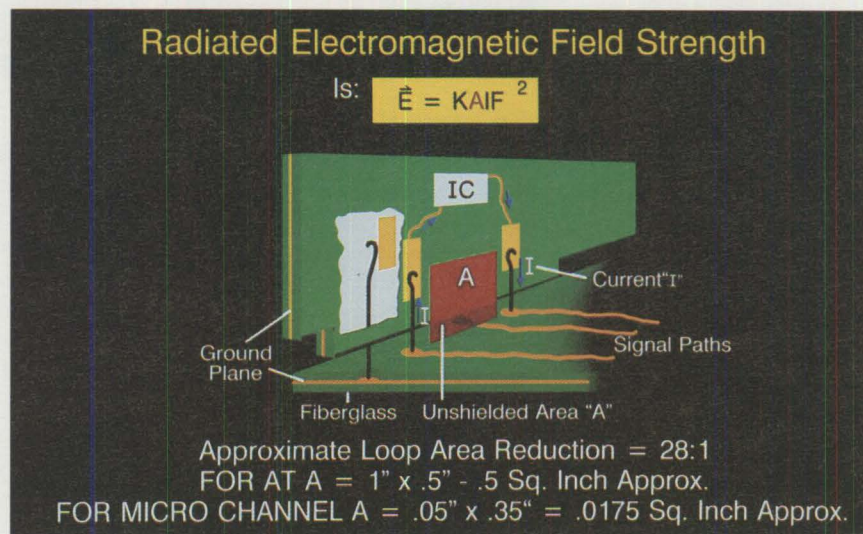


Figure 16. Radiated Electromagnetic Field Strength

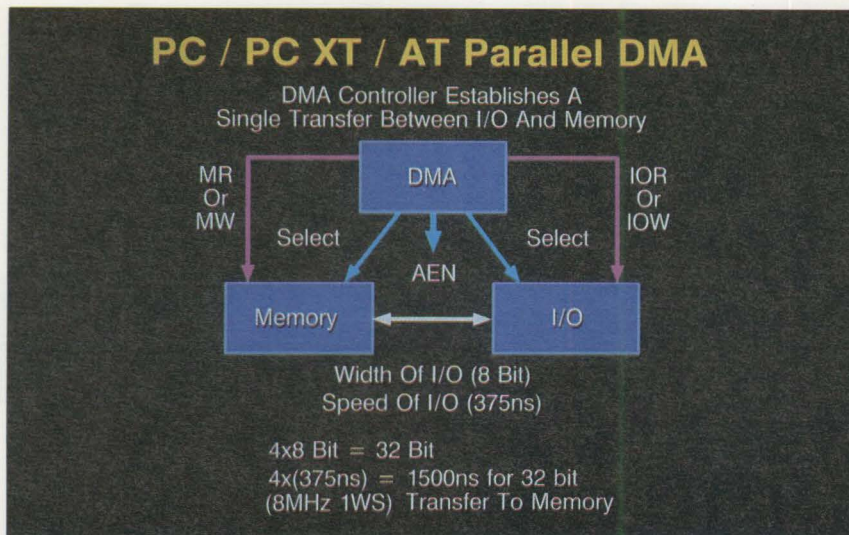


Figure 17. PC/XT/AT Parallel DMA

plementations, that moves data with just one operation.

First, let's understand the way the PC, PC XT, and AT handle a DMA transfer (Figure 17). The PC DMA controller does not actually transfer data. It is a third party that arranges a transfer out from a source (for example, memory) and into a destination (in this example, an I/O adapter). The transfers occur as one operation, in parallel, hence the name "parallel DMA."

The parallel transfer begins when the DMA controller selects the memory by address, selects the I/O adapter by a signal called "DMA acknowledge," and asserts a signal called "AEN." AEN says to all the non-DMA adapters in the system, "I will drive two data strobes at once," in this case, memory write and I/O read. Only the parties in the DMA transfer respond, and data moves out of memory, across the data bus, and into the I/O adapter, all as one operation.

If everything occurs as one operation, then the width of the transfer

must be the width of the narrowest party. That is, an 8-bit I/O device could not receive 32 bits of data from a 32-bit memory in one operation; it would take four operations, each 8 bits wide, to move the 32-bit value. The most common I/O width is 8 bits, and today most systems are being defined for 32-bit memory width, so this is a realistic example. Likewise, the duration of the transfer must be the length of the slowest party. Here again, it is the I/O adapter on the AT bus that moves data at the default speed of the AT bus — 375 nanoseconds (ns), or 375 billionths of a second. It will take four 375 ns, 8-bit-wide transfers to move a 32-bit quantity of data from memory to I/O, or 1500 ns. The memory and the I/O are both delayed 1500 ns to complete the four operations.

Now let's understand serial DMA as it is used in the "string mov" operation of the 80286, 80386, and 80486 processors and by Micro Channel DMA as well (Figure 18). In a Micro Channel DMA controller, a register is directly involved in the transfer. Data is read into the

register in one or more operations and written to a destination with one or more operations. The transfers occur sequentially, hence the name "serial DMA."

Here again, four 8-bit transfers would be required to move a 32-bit value from an 8-bit I/O adapter to the DMA controller, and at least one more to read the data out. There are still more operations than with PC-, PC XT-, and AT-style parallel DMA, but they occur much faster on the Micro Channel interface. The four 8-bit transfers can occur at the default speed of the Micro Channel interface, 200 ns each. The total time is 800 ns.

The transfer with the I/O adapter was at its optimal speed, and the transfer with memory can be at the optimal speed of a 32-bit cached storage (as used in the Model 70-A21). Assume an average duration of 50 ns. If the operations occur on one bus, the total duration is 850 ns, or faster by just slightly over half the time of the AT bus. If the memory is on a separate bus, the I/O transfers do not impede its operation, and the serial DMA transfer with memory will occur 30 times faster than the AT-style parallel mode.

Greater Latitude for System

Design: Whether a computer is designed with one bus or more than one bus is a function of the system organization, but not of the channel architecture. Cost-effective Micro Channel systems can be defined with all the memory and I/O on one bus, as in the PS/2 Models 50 and 60. In addition, systems with the memory on one bus and the I/O on another, such as the PS/2 Model 70-A21, can also be defined with the same Micro Channel specification. It is important that high-function, ad-

vanced architecture does not imply higher complexity and cost – and it does not with Micro Channel architecture. The business case for card development is built on the large number of sockets installed for a wide range of products and not on the limited number of sockets for high-end systems alone.

One thing that allows this flexibility is that each of the parties in the system moves data at its individual optimal rate rather than in lock step with a system processor. Asynchronous transfer allows the straightforward Micro Channel

implementation of processors of different types and speeds than the one that may have been soldered to the system board at the time of purchase.

Full Specifications: A full specification of all channel timings, current and capacitance loadings, mechanical and electrical considerations, and full definition of the functions are available in the *IBM Personal System/2 Hardware Interface Technical Reference* manual for the PS/2 Models 50, 60, 70386 and 80386 computers.

Note: IBM regularly holds education sessions and offers design assistance to adapter card developers.

Micro Channel Reliability: The Micro Channel interface has significantly improved the reliability, availability and serviceability of the system unit as well. Micro Channel design assists in the implementation of integrated circuits (ICs). IC designers most often run out of pins in the package before the silicon circuitry inside is completely used. More information is packed into fewer signals with the Micro Channel interface, and fewer pins are required to define the same function

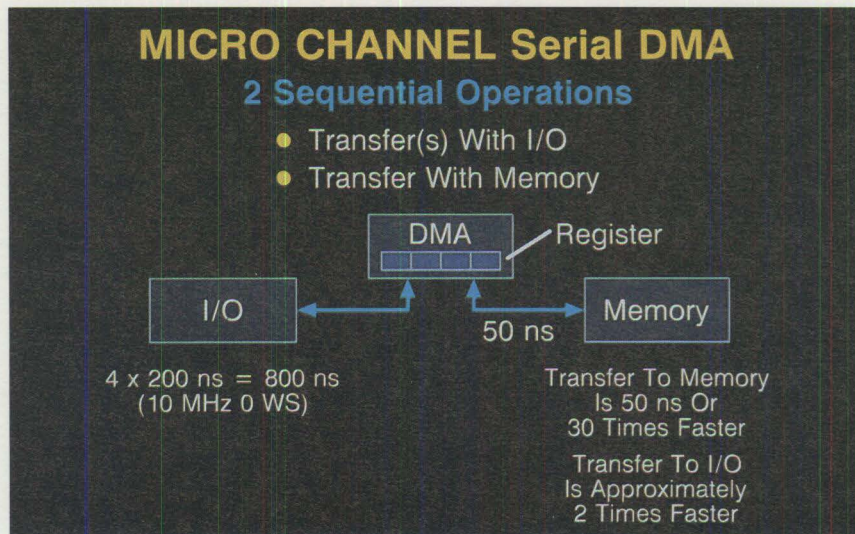


Figure 18. Micro Channel Serial DMA

in an IC. A larger portion of the circuitry in a system can be included inside the more reliable silicon chip, and the reliability of the entire system increases.

Because the Personal System/2 Model 50 (Figure 19) uses integrated circuits, it has one-eighth of the board area of the AT 339 that it replaces, is 25 percent faster, and includes six I/O adapters on the planar. It has a mean time between

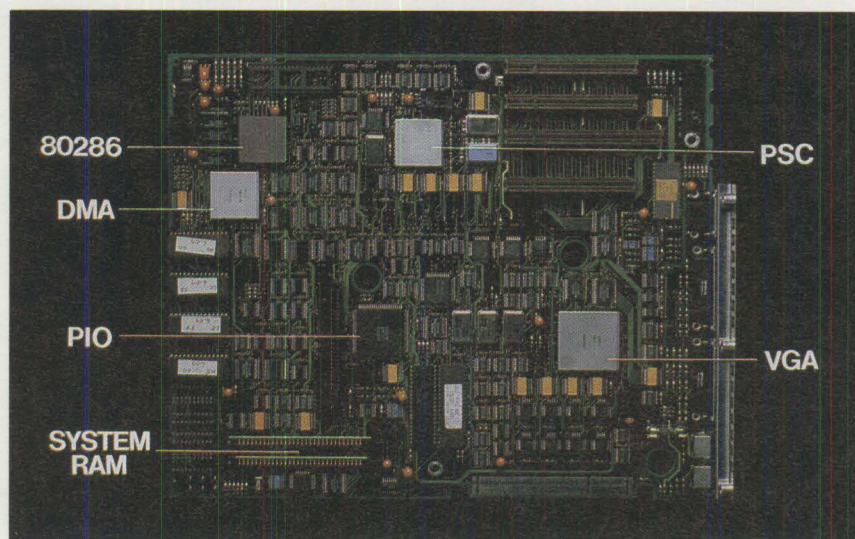


Figure 19. PS/2 Model 50 System Board
 80286: Intel 80286 Processor
 VGA: Video Graphics Array
 DMA: Direct Memory Access Chip
 PSC: Processor Support Chip
 PIO: Planar (System Board) I/O Chip

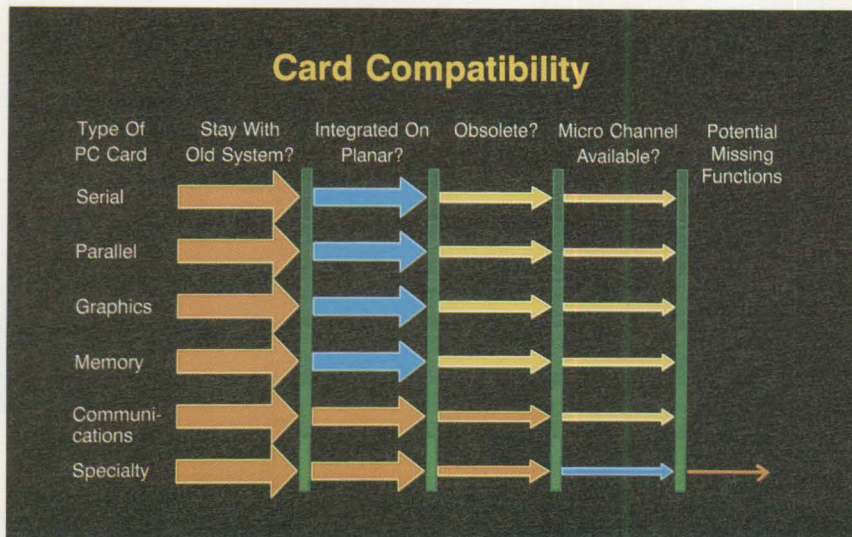


Figure 20. Card Compatibility

failure (an index of reliability) that is two to three times better than that of the AT 339 system, and has a service cost that is approximately one-third that of the AT 339. It also consumes only 94 watts, or half the power of most AT-based systems. These are all advantages to the customer.

Micro Channel architecture was planned from the ground up to support tomorrow's applications. It has the data integrity and reliability required of a system that multiple users may depend on.

Apples and Oranges: We often see comparisons between what Micro Channel architecture was in its first 16-bit implementation, and the maximum configuration of what implementation of another architecture might offer in the future. The AT computer might be extendable, just as the Micro Channel computer will be expanded. While we will not describe what extensions will be made to Micro Channel architecture here, one comparison between today's implementations is often not recognized: the DMA transfer rate

for 16-bit operations on the Micro Channel bus is often 10 times the DMA transfer rate of AT systems that employ 5 MHz Intel 8237-compatible DMA controllers.

Compare the what-might-be cases – for example, what if the AT and Micro Channel DMA controllers each were implemented as bus masters. In both cases, the maximum burst rate of the Micro Channel bus would be nearly four times that of the AT bus – with both buses operating at default rates. This is because a system that exchanges data with existing AT cards could move two bytes in 375 ns. The Micro Channel interface defines 4-byte transfers between master and slave in 200 ns. The 32-bit SCSI interface bus master has been implemented and demonstrated with 4-byte transfers every 240 ns, or nearly 17 million bytes per second. This approximates the calculated continuous rate of 18.7 MB. That is approximately five times the rated speed of the AT bus as given in other manufacturers' specifications.

As for claims that lack of AT card compatibility limits a system designer's options, many of the cards in systems today must stay right where they are because the diskette, fixed disk, display, and memory cards are matched to the data width, speed, and peripheral interfaces of those systems (Figure 20). If the cards are removed, the system peripherals will not work.

Most of these interfaces have been upgraded. Where they are implemented on the PS/2 system board, no card need be purchased at all. Furthermore, many of the communications and other functions, purchased for the old system when it was new, are now obsolete, and more current designs are now available for the PS/2 Micro Channel computers. It is important to remember that all Micro Channel cards are current – with none made before April 1987.

Card Portability: Studies have shown that it is indeed rare for a user to "cannibalize" a system to recover even a few cards for installation in a new system. Businesses need the most current and productive systems to be competitive. In the workplace, the old system is typically given to a new user whose needs are met by the capability of the old system. Almost 98 percent of the time, all the cards in a system remain in that system. Little more than 2 percent of the cards are brought forward (Figure 21).

LAN cards are the cards most frequently drafted into new duty. It is important to note that this data was obtained before Ethernet® LAN adapters were widely available for Micro Channel systems. Today there are several Micro Channel Ethernet bus master cards, and as many slave adapters, available.

There is a similar situation for 3278 emulation boards. Token-Ring LAN has recently gone through a generational change from 4-megabit to 16-megabit interface as well.

Today there are over 1,000 Micro Channel adapter cards available. IBM has issued more than 2,700 design IDs worldwide, indicating that as many as 1,700 more cards are under development. More than 650 vendors have the Micro Channel standard in their plans today. Of the 1,000 cards, 27 are advanced bus master cards; 197 bus master IDs have been assigned. Indeed, this is a better story for Micro Channel advanced I/O card designs than for AT cards.

Dollars and Common Sense

It's easy to get excited about advanced technology, but dollars and common sense are what influence most business decisions. The reason to buy Micro Channel architecture is that it comes for free, in the highest-quality personal systems available today - the PS/2 Models 50 and above. They are also among the least expensive to own!

Assume you intend to own your new system for five years. Let's analyze the cost of ownership. We'll use the AT 339 system for comparison, but data for any other AT-bus machine can be used instead.

During the first year, both systems are under warranty, so the service cost is identical - zero. Based on IBM's annual onsite maintenance charges as of July 1, 1989, years 2 through 5 show that the Micro Channel system has a much lower IBM service cost. This is a reflection of higher reliability and more efficient diagnostics, as well as a reduction

in "no trouble found" service calls due to mis-set switches.

Diagnostics in Micro Channel systems can help prevent costly and unnecessary replacement of functional components due to a greater ability to accurately isolate and replace only the failed components; this can further reduce the service cost.

When you upgrade the system display, you do not need to pay a second time for the VGA function as you would have to in an AT system. This is because both the input and output of the display controller are connected to the Micro Channel interface. Cards that extend the display capabilities do not need to discard the existing controller or replace the function on the extension card. This can save the original VGA investment that need not be added to cost of the extension display adapter. This amounts to a savings of several hundred dollars.

PS/2 system users also won't have to spend as much on electric power. Check the name plate power rating on the back of most AT-based sys-

tems. The AT 339 is 192 watts, and, fully configured, it can cost you \$650 to run it five years for 24 hours per day, 292 days per year, at 10 cents per kilowatt hour. Adjust this number if you use the machine more or less, or if your local power costs more or less.

While some may say that the lower power allotted to adapter designs in Micro Channel systems makes the designer's job more challenging, those challenges can be overcome by large scale integration (LSI) and computer-aided design techniques that reduce size and power consumption and make the design less expensive to reproduce and more reliable in operation. The reduced power in PS/2 Micro Channel systems is an advantage to the customer. The name plate rating for a maximally configured PS/2 Model 50 Z is 94 watts, about one half that of the AT system it replaced. The result: you'll save about \$325 in electric bills. (The display also uses electricity but is not included in this system unit comparison.)

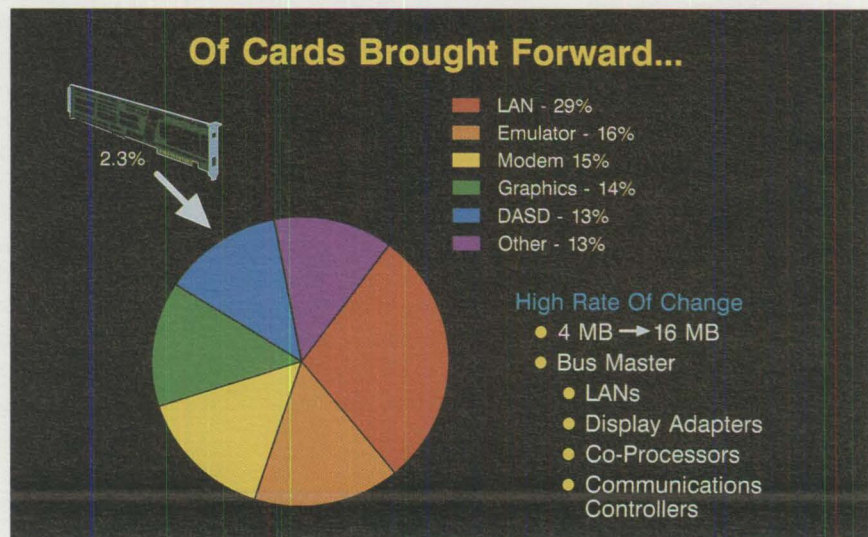


Figure 21. Cards Brought Forward

Add up all the savings you can demonstrate, and then consider comparative depreciation in the resale value of the system, and the value of increased productivity from a system that is more available and needs less maintenance time. You'll find you save a sizable fraction of the purchase price. Your calculations would look something like this:

On-Site Maintenance:

Year	AT 339	PS/2 50 Z	You Save
1	0	0	0
2	442	192	250
3	442	192	250
4	442	192	250
5	442	192	250
Display Upgrade	400	0	400
Electricity	650	325	325
Total			\$1,725

In fact, if you keep the system a few more years, it will have been free – the savings in ownership will have paid for the original purchase.

The moral is to look past the cost of purchase to the real cost of ownership before you decide which system offers the lowest overall cost. This is why Micro Channel architecture also stands for "Maintain the Customer's Assets" – maintain the asset in the system unit by building the future in and "designing in" quality, and maintain the asset in the feature cards by assisting in the portability of those functions across a broad range of machines.

Evolution Follows Revolution

The future will see the Micro Channel standard grow and evolve grace-

fully. New functions are already defined for advanced systems. They will include further improvement in the ability of the system to detect, in some cases to recover from, and perhaps even to correct, errors.

Although implementations of the Micro Channel interface have already defined data transfers at 32 megabytes per second, the demands of multitasking and multiuser systems will further increase throughput requirements. By adjustments in the efficiency of transfer protocols and wider-width data transfers, we are confident that the throughput can be significantly increased over time.

The initial Micro Channel specification defined 32-bit DMA, and it will appear in future systems. Most exciting of all, control block architectures, not unlike those in use on IBM minicomputers and mainframes, are enabled by the Micro Channel bus master definition. Looking many years ahead, the future has been built into these systems by the definition of far-reaching standards – standards that help complex systems to be defined and sourced from multiple manufacturers, and integrated with confidence of sustained trouble-free operation.

A large infrastructure now supports the Micro Channel standard interface, and that trained design base is growing. Micro Channel architecture can grow into the future because it is part of a system of standards, and a vision that does the same with the documented inter-

faces for Advanced BIOS and Systems Application Architecture™ (SAA™), the operating system's interface to applications.

Micro Channel architecture is simply an element of a structure. Philosophically it is not just an extension of the now dated PC bus for advanced microcomputer systems. It is indeed the opposite – Micro Channel architecture incorporates mainframe architecture principles, innovatively made cost-effective and compatible for advanced microsystems.

ABOUT THE AUTHOR

Chet Heath is a senior engineer at IBM's Entry Systems Division laboratory in his twentieth year with IBM. He holds a bachelor of science degree in electrical engineering from the New Jersey Institute of Technology and a master of science degree in electrical engineering from the IBM LSI Institute at the University of Vermont. He refers to himself as "the oldest living survivor of Micro Channel architecture" because he was involved in its definition since inception and gave the architecture its name. He has received IBM Quality, Outstanding Technical Achievement, Outstanding Innovation, and Corporate Technical Achievement Awards. Chet also has attained the eighth level of invention awards. He is presently assigned to development of Micro Channel strategic enhancements.

Micro Channel System Configuration Considerations

*Carl Grant, Don Ingerman
and Robert Robinson
IBM Corporation
Boca Raton, Florida*

This article addresses Micro Channel system configuration considerations to enable those who are responsible for selection of Micro Channel products to understand the basic systems and adapter structures, the basic system performance parameters, and the basic configuration considerations to make intelligent choices in the selection of Micro Channel systems and feature adapters.

The capabilities of personal computers have increased significantly over the last several years. During this same period, there has been a dramatic change in the environments in which personal computers are used.

When first introduced, the personal computer was exactly that, a computer intended for the personal use of a single user. Response times were related to the perception of the person sitting at the keyboard. Word processing, for a single user, was one of the primary applications. In this environment, there was little, if any, need for overlapping operations. The input/output (I/O) for these systems was normally handled by what is called programmed I/O. In this mode, when an operation was initiated to or from an I/O adapter (for example, a keyboard, display or file), a special program took control of the system processor to handle the movement of the data to and from the adapter. The appli-

cation program in progress was suspended until the I/O operation was completed. Thus, the I/O operation and the application execution were serial, not overlapping. This type of structure is called single-tasking.

It soon became apparent that some overlapping of operations would improve the productivity of the system and the user. An example of this is the ability to print one file while updating a second file. One way to support this type of operation is to restructure the software running on the system so that it can support multiple tasks running at the same time. This type of structure is called multitasking. More efficient handling of I/O operations was needed to realize the full benefits of the multitasking operating environment. A new method of controlling I/O, direct memory access (DMA), which frees the system processor

from being involved in the movement of data, was introduced.

New uses for personal computers have rapidly developed due to improved performance and capabilities of the new systems and the availability of multitasking operating systems. Personal computers are now commonly used as file servers, print servers, network nodes and in other applications that require the simultaneous support of multiple users. This type of operation is called multiuser. An increased emphasis on the ability of the overall system to provide a consistent level of satisfactory performance for all users is now required. This in turn has resulted in further changes in the interface between the system processor and its attached I/O adapters. To reduce the amount of time that the I/O adapter is using the I/O bus, the data bus width has been increased



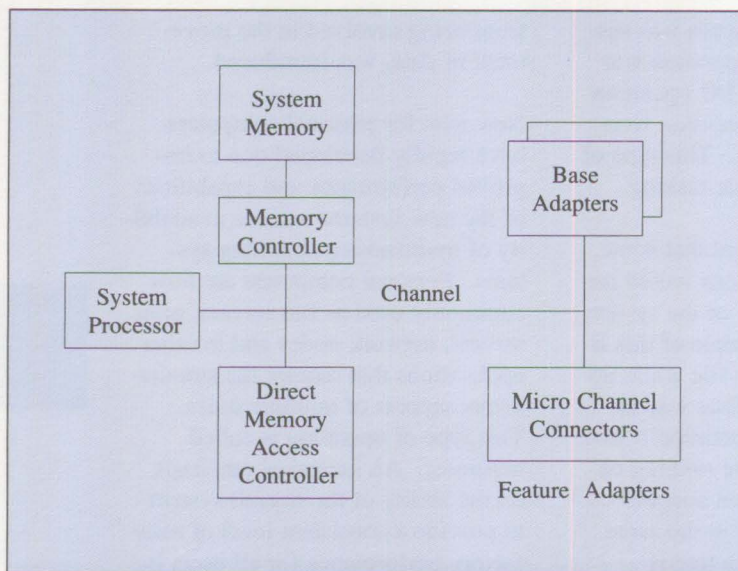


Figure 1. System Hardware

and the data transfer time has been decreased. More intelligence is incorporated in the adapters, further reducing the amount of control that is required from the system processor. Micro Channel architecture allows the adapter to become the bus master. In this mode, once the adapter has gained control of the channel, it controls all of the activity associated with the data transfer, without impacting the system processor.

New systems structures have been introduced that improve overall performance in the multitasking environments. An example of this is the memory cache that has been implemented (in some systems) between the system memory and the system processor. The addition of this memory cache reduces the potential contention between the system processor and the adapters for access to system memory. This results in improved performance by the system processor and by the adapters attached to the system.

These new levels of sophistication in personal computers make it more important to select the proper components to obtain a balanced and efficient system. There are a large number of adapters available for use on Micro Channel systems. Many of these adapters perform similar functions. In order to select the best adapter for a given system and its applications, not only the function provided by an adapter must be considered, but also how that adapter will perform in the total system.

The purpose of this article is to present a discussion of the considerations that are necessary in selecting components for a Micro Channel system. These include system hardware and feature adapters.

Micro Channel System Hardware

The Micro Channel system hardware consists of the:

- System processor
- Channel
- Memory subsystem, which includes the memory controller and the system memory
- Micro Channel connectors
- Direct memory access (DMA) controller
- Adapters
- System board
- Support logic required for basic operation that is furnished with the system board

This structure is shown in Figure 1.

System Structure and Components

The basic system structure consists of the:

- System processor
- Processor bus
- Memory bus
- I/O bus, called the *channel*
- System memory

This structure is shown in Figure 2.

The processor bus, the memory bus and the channel are the same logical path. Therefore, both the system processor, for instruction and data fetching, and the adapters, for data transfers, use the same resource.

An enhanced system structure has been introduced to improve overall system performance. This enhanced system structure places a high-speed memory, called a *memory cache*, between the system processor and the system board memory. Figure 3 shows that for this structure, the memory cache is connected to the system processor by the processor bus, and to the system memory by the memory bus. The processor bus is isolated from the memory bus/channel.

When instructions or data are fetched from the memory cache, called a *cache hit*, the processor bus is used and transfers can be occurring simultaneously on the memory bus/channel. When the instructions or data required by the system processor are not in the memory cache, called a *cache miss*, the information must then be fetched from system memory using the memory bus. When a cache miss occurs, the system processor may contend with an adapter data transfer for the memory bus/channel.

The enhanced system structure offers two advantages. The first advantage is that when instructions or data are fetched from the memory cache, it takes less time to fetch this information because of the faster memory access times. The second advantage is that when information is fetched from the memory cache, the memory bus/channel is not used, and an adapter data transfer can overlap the system processor instruction execution. When there is a cache hit, the potential for contention with an adapter data transfer is eliminated.

System Memory

System memory is memory addressable by the system processor mem-

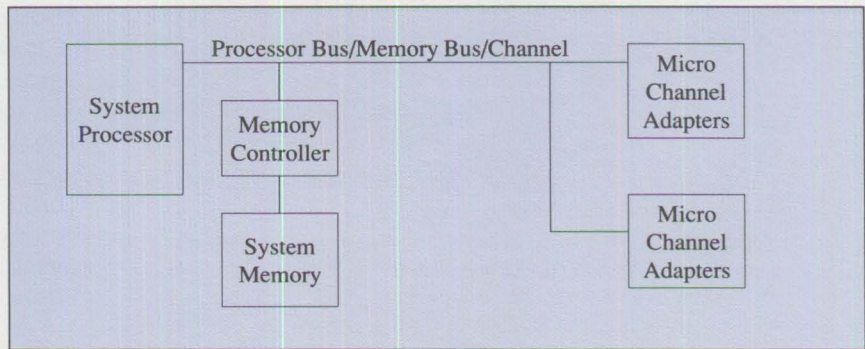


Figure 2. Basic System Structure

ory address space. The system software and data structures are resident in system memory. The system memory furnished with the system is resident on the system board.

In an IBM PS/2 computer system, the system processor memory address space and the Micro Channel memory address space are a single address space. This allows system memory to be resident on an adapter that occupies a Micro Channel connector. System memory that resides on an adapter is called Micro Channel system memory. This offers the option to use Micro Channel system memory to expand, or to upgrade, system memory beyond the capability provided by system board memory.

Adapters

Adapters control the additional functions that are needed to give the system the desired functional capability. These adapters are known as base adapters, if they are furnished as part of the system, or feature adapters, if they are purchased separately. Base adapters enhance the basic system with generally required functions (for example, video graphics array, serial port, parallel port, keyboard controller, diskette controller, fixed file adapter). Base adapters are either on the system board or in a Micro Channel connector. Feature adapters are inserted into Micro Channel connectors. Figure 1 shows base adapters and feature adapters in relation to the rest of the system hard-

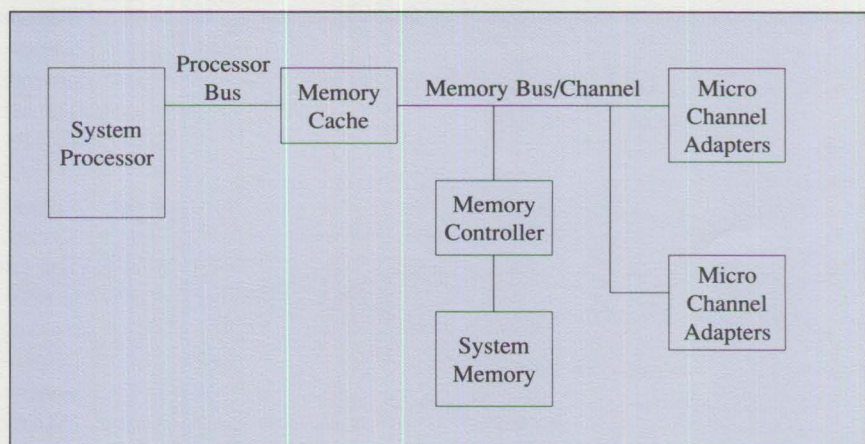


Figure 3. Enhanced System Structure

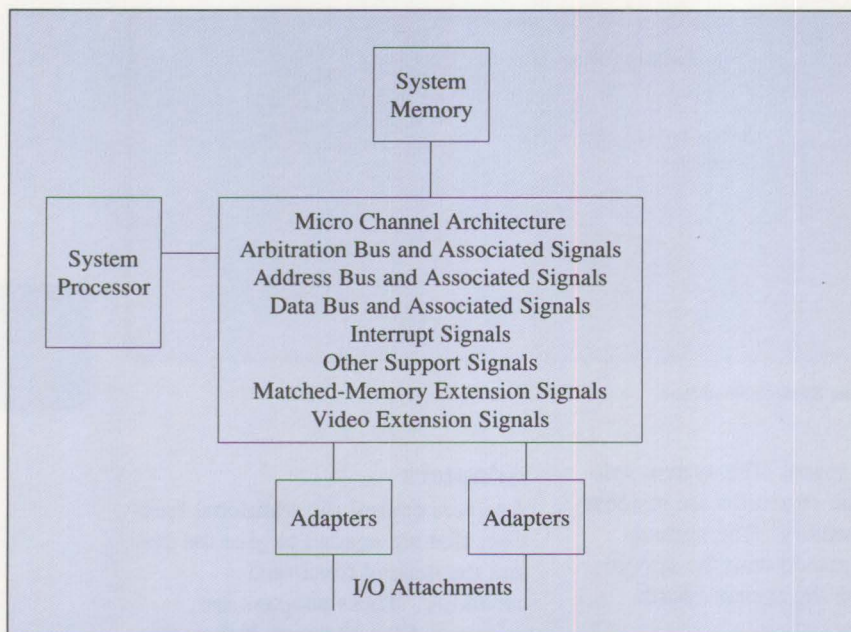


Figure 4. Micro Channel Structure

ware. Examples of feature adapters include coprocessor adapters, serial port adapters, memory expansion adapters, host communications adapters, and network adapters.

This article discusses the adapter attachment interface to the channel. Devices attached to the non-channel side of an adapter, for example, a

printer attached to a serial port adapter or a DASD attached to a SCSI adapter, and the related protocols are not included.

Micro Channel Architecture

Micro Channel architecture provides a high-speed data highway that connects the system processor, the system memory and the adapters.

Micro Channel architecture consists of an arbitration bus and associated signals, an address bus and associated signals, a data bus and associated signals, interrupt signals, other support signals, and optional extensions for matched-memory extension signals and video extension signals. This structure is shown in Figure 4.

Micro Channel addressing consists of two separate address spaces: a memory address space and an I/O address space. The address bus and associated signals provide either a memory address or an I/O address on the channel (see Figure 5).

The I/O address space consists of 64 KB (KB = 1024 bytes) I/O addresses. During an I/O cycle, the I/O address space is addressed by 16 bits of the address bus.

The memory address space can be as large as 4 GB (GB = 1,073,741,824 bytes). During a memory cycle, the memory address space is addressed by the address bus. A 24-bit address can address 16 MB (MB = 1,048,576 bytes) of memory and a 32-bit address can address 4 GB of memory.

Arbitration Bus and Associated Signals:

Micro Channel architecture provides a hardware arbitration procedure to control bus ownership. The arbitration bus and associated signals are used in the arbitration procedure, which prioritizes and resolves up to 16 requests (arbitration levels), for control of the channel. The winner of the arbitration is the temporary bus owner, called the *controlling master*. The arbitrating participants can be adapters or the system processor (see the section, "Micro Channel Participants"). The system processor is assigned an arbi-

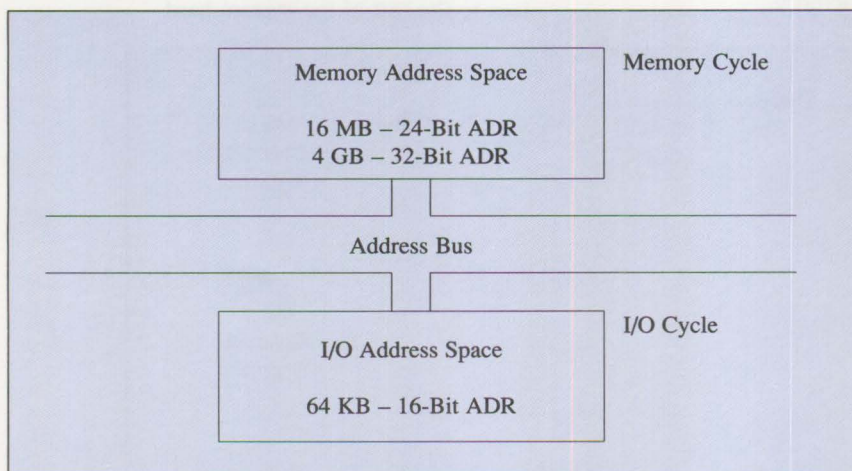


Figure 5. I/O and Memory Address Space

tration level, leaving 15 arbitration levels available for adapters.

The arbitration procedure supports a burst mode. Burst mode allows an adapter to retain control of the channel to perform multiple data transfers without arbitrating for the channel for each data transfer. The data transfer continues until the data transfer is complete, or for up to 7.8 microseconds after another arbitrating participant requests the channel. At this point, channel ownership is released. The arbitration procedure is serial with data transfer, either an arbitration procedure or a data transfer can occupy the channel at any one time. Therefore, the use of burst mode provides for the maximum efficiency in the use of the Micro Channel bandwidth.

Address Bus and Associated

Signals: The address bus and associated signals are used by the controlling master to select a slave during a data transfer procedure. Either an I/O space address or a memory space address is indicated. The address, on the address bus, is also used to select the storage location for the data transfer.

Data Bus and Associated Signals:

The data bus is used to transfer either 8, 16, 24 or 32 bits of data. The associated signals indicate the width of the transfer and the direction of the data transfer, read or write.

Interrupt Signals: A request for processor attention, called an *interrupt*, is presented by an adapter through an interrupt signal. Each interrupt signal is on a system interrupt-priority level. The adapter request for service is processed by the system processor at the requested priority. Micro Channel architecture supports 11 interrupt

levels. To allow configuration flexibility, each interrupt level may be concurrently shared by multiple adapters.

Other Support Signals: The other support signals include the capability for integrating a single-channel audio signal on the channel. Adapters may exchange and process audio information using this audio signal.

Matched-Memory Extension

Signals: Matched-memory extension signals support a unique data transfer procedure between the system processor and its channel-resident memory (see the section, "Matched-Memory Extension").

Video Extension Signals: The video extension signals provide the capability for a Video Graphics Array (VGA) feature adapter to transfer video data between the video subsystem and the feature adapter. For example, the IBM PS/2 Display Adapter allows upgrade of VGA performance beyond that provided by the base VGA adapter.

Micro Channel Participants

The system processor, system memory and adapters attached to the channel contain entities called *participants*. This section describes the Micro Channel participants. See Figure 6 for a diagram of the participants and the data transfer paths supported.

The Micro Channel participants are either masters or slaves. A master, or a DMA slave, arbitrates for access to the channel and supports data transfer to or from a slave. A slave sends and receives data under the control of a master, called the controlling master.

Masters: There are three types of implemented masters: the system processor, called the *system master*, the DMA controller and a bus master. The system processor and the DMA controller are provided with the system hardware. The bus master is an adapter implementation.

System Master (System Processor):

The system processor is the processor provided with the system hardware. The system processor

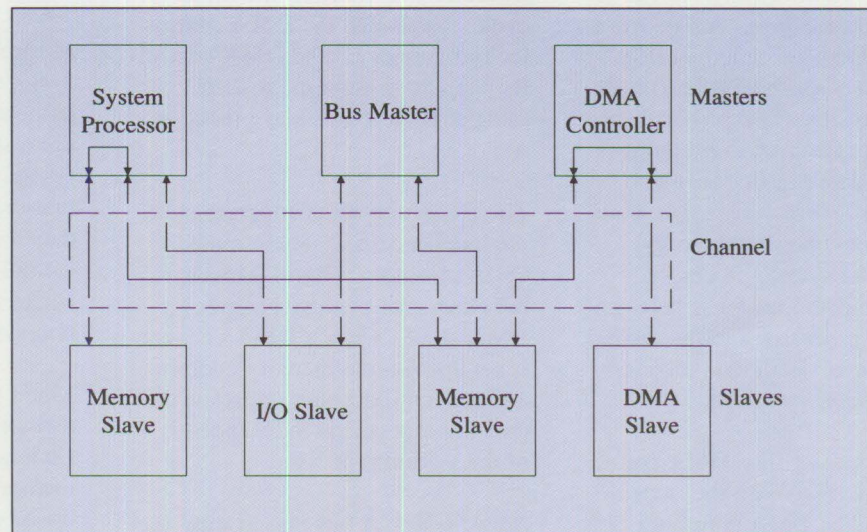


Figure 6. Micro Channel Participants and Data Transfer Paths

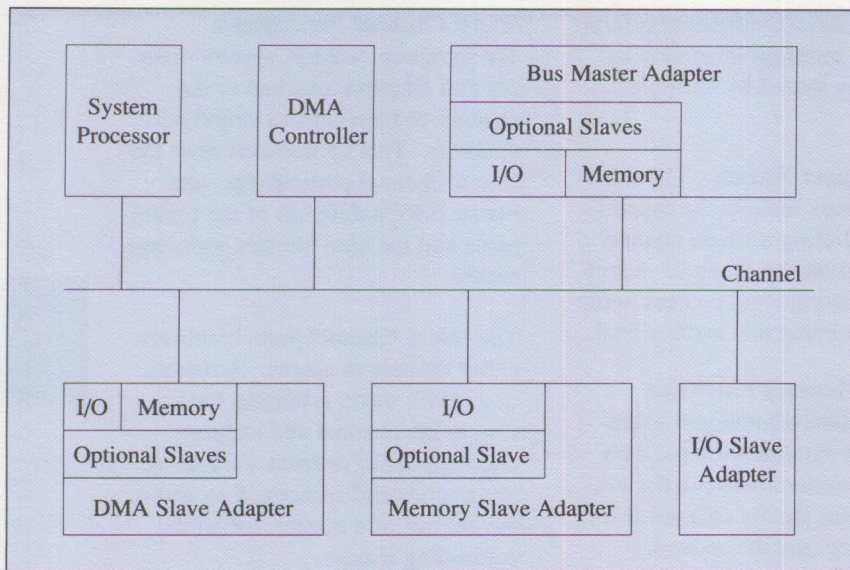


Figure 7. Adapter Participants

controls and manages the system configuration. It may initiate a request for use of the channel. The system processor owns the channel when no other master owns the channel; it is also known as the *default master*.

The system processor supports data transfers to or from an I/O slave or a memory slave. The data transfers are directly controlled by system processor instructions. A data transfer to an I/O slave, called an *I/O cycle*, is accomplished in one physical data transfer. Two physical data transfers are required to accomplish one actual data transfer between two memory slaves. One data transfer is between the first memory slave and the system processor. The second data transfer is between the system processor and the second memory slave. Both data transfers are called *memory cycles*.

DMA Controller: The DMA controller is provided with the system hardware. The DMA controller does not arbitrate for the channel.

A DMA slave that requires a data transfer will perform the arbitration.

The DMA controller manages the transfer of data between a DMA slave and a memory slave. This operation requires two physical data transfers to accomplish one actual data transfer. One data transfer, the I/O cycle, is between the DMA slave and the DMA controller; and a second data transfer, the memory cycle, is between the DMA controller and memory. The DMA controller supports a burst mode data transfer if the DMA slave requests it.

The DMA controller supports multiple independent channels, called DMA channels. Each DMA channel allows for the attachment of a DMA slave. Once a DMA channel is set up with the memory address and the transfer count, in bytes, the data transfer occurs independently of the system processor.

Bus Master: A bus master arbitrates for the channel and manages the data transfer to or from an I/O

slave or a memory slave. The data transfer is accomplished in one physical data transfer. The bus master can, optionally, support a burst mode data transfer.

A bus master can receive an interrupt signal from an I/O slave adapter. Either the system hardware or a bus master will be enabled to receive an interrupt level, but not both. It is not possible for two masters to receive the same interrupt. Therefore, the interrupt level that is received by the bus master must be disabled in the system hardware by a software procedure before it can then be enabled in the bus master.

Slaves: There are three types of slaves: I/O slave, memory slave and DMA slave.

I/O Slave: During an I/O cycle, an I/O slave is selected by an address in the I/O address space. The system processor or a bus master is required to perform the data transfers. Either the master will poll the adapter to initiate the data transfer or the I/O slave will interrupt the master to initiate the data transfer.

Memory Slave: During a memory cycle, a memory slave is selected by an address in the memory address space. A master (system processor, DMA controller, or a bus master) is required to perform the data transfers. The two basic applications of a memory slave are Micro Channel system memory and non-system memory.

Micro Channel system memory is memory that is installed in a Micro Channel connector. The system memory is assigned a range of addresses within the Micro Channel memory address space. The Micro Channel system memory may be

used as shared memory among adapters or as instruction memory for a bus master adapter. For an IBM PS/2 computer system, the Micro Channel system memory also can be used to expand system memory beyond that supported by the system board (see the section, "System Memory").

Non-system memory is memory in the Micro Channel memory address space that is on an adapter. The non-system memory is used for data transfer between the memory slave and a controlling master. This is referred to as memory-mapped I/O.

DMA Slave: A DMA slave arbitrates for the channel. The DMA slave relies on the DMA controller to be the controlling master that manages the data transfer between the DMA slave and a memory slave. The DMA slave is associated with a DMA channel in the DMA controller. The DMA slave can, optionally, support a burst mode data transfer.

Participant Usage

The functional capability of one or more Micro Channel participants is incorporated into an adapter. The primary mode of operation (*operational mode*) of the adapter determines its primary participant when there is more than one participant. The adapter type is the same as its primary participant. There are four basic adapter types: an I/O slave adapter, a memory slave adapter, a DMA slave adapter, or a bus master adapter (see Figure 7). If an adapter performs a data transfer between an I/O device and memory using the functions of a DMA slave participant, it is a DMA slave adapter. If an adapter performs a data transfer between an I/O device and memory using the functions of a bus master participant, it is a bus

master adapter. In the selection of an adapter, see the section, "Adapter Participant," for the adapter performance considerations.

As shown in Figure 7, an adapter may incorporate one or more secondary participants in addition to the primary participant. The secondary participant can be used to support adapter initialization and control functions required in the execution of its operational mode. A second usage is adapter to adapter data transfer between two bus master adapters (see the section, "Adapter-to-Adapter Data Transfer"). The controlling bus master performs the data transfer to a memory slave participant in the second bus master adapter. The operational mode of both adapters is a bus master.

The adapter system resource requirements (for example, interrupt level, DMA channels, arbitration levels, I/O address space addresses, and memory address space addresses) are the total of the primary participant requirements and any additional requirements of one or more secondary participants.

Address Bus Support

A Micro Channel adapter implements a 16-bit, 24-bit or 32-bit address bus. Figure 8 provides the currently supported address bus widths for each participant.

Data Transfer

Micro Channel data transfers can be between the system processor and an adapter, the system processor and system memory, an adapter and system memory, or an adapter-to-adapter. The significant performance parameters are the data bus width and the data transfer time, which yield the data transfer rate.

Data Bus Width: A Micro Channel adapter can implement an 8-bit, 16-bit or 32-bit data bus width. For example, a 32-bit data bus means that the data transfer width can be up to 32 bits, and that each data transfer can be 8 bits, 16 bits or 32 bits. The data transfer width is equal to the lesser of the data bus width of the controlling master and the selected slave, or of the data width of the selected I/O address (it can be less than the data bus width).

Figure 8 provides the currently supported data bus widths for each participant.

Data Transfer Time: The data transfer time is determined by the data transfer procedure and the capability of the two participants involved in the data transfer. The data transfer time is the greater of the data transfer times supported by the controlling master and the selected slave.

Basic Data Transfer Procedure:
Micro Channel architecture supports

Participant	Data Bus Width	Address Bus Width
System Processor	16-bit or 32-bit	24-bit or 32-bit
DMA Controller	16-bit	24-bit
Bus Master	16-bit or 32-bit	24-bit or 32-bit
DMA Slave	8-bit or 16-bit	16-bit
I/O Slave	8-bit, 16-bit or 32-bit	16-bit
Memory Slave	8-bit, 16-bit or 32-bit	24-bit or 32-bit

Figure 8. Participant Data Bus and Address Bus Width

Transfer Width	Transfer Time		
	200 ns	300 ns	3800 ns
8-bits	5 MB/second	3.3 MB/second	0.26 MB/second
16-bits	10 MB/second	6.6 MB/second	0.52 MB/second
32-bits	20 MB/second	13.3 MB/second	1.05 MB/second

Figure 9. Data Transfer Rate

a range of data transfer times for a basic data transfer procedure. The default data transfer time is a minimum of 200 nanoseconds. Either the system or the adapter may extend the 200 nanosecond minimum time. An adapter may implement an extended data transfer time from 300 nanoseconds to 3800 nanoseconds.

Matched-Memory Extension: A matched-memory extension is a unique data transfer procedure between the system processor and the selected Micro Channel system memory adapter. The procedure provides for a data transfer time that is less than the default data transfer time of that system. Both the system hardware and the Micro Channel system memory adapter must support the matched-memory extension, or the data transfer will be performed with the basic data transfer procedure.

Data Transfer Rate: The instantaneous data transfer rate of an adapter on the channel is determined by the data transfer width and the data transfer time. Figure 9 provides the range of capability of the Micro Channel architecture for the basic data transfer procedure.

The DMA controller furnished with IBM PS/2 computer systems has an 16-bit data bus width capability that is independent of the system data bus width. As two physical data transfers are required to accomplish

one actual data transfer (see the section, "DMA Controller"), the maximum data transfer rate is limited to 5 MB per second (MB/sec) for a DMA slave adapter with a 16-bit data bus, or 2.5 MB/sec for a DMA slave adapter with an 8-bit data bus. (This does not include the enhanced functions described in the article, "Overview of Extended Micro Channel Functions" in this issue.)

Adapter-to-Adapter Data

Transfer: Micro Channel architecture supports an adapter-to-adapter data transfer that does not involve system board memory or the system processor. The transfer is between a bus master adapter and a memory slave or an I/O slave adapter, or between a DMA slave adapter and a memory slave adapter through the DMA controller.

A bus master adapter may optionally support data transfer to or from another bus master acting as a slave. The controlling bus master initiates the data transfer to or from the selected memory slave participant supported by the other bus master. Either bus master may be the controlling master, providing a symmetrical data transfer capability. This method of data transfer is referred to as *peer to peer*.

Buffered Adapters

A *buffer* is storage in an adapter that is used to temporarily hold data. The purpose is to compensate for the difference in rate of flow of data or time of occurrence of

events, when transferring data between the adapter and memory. The buffer permits the transfer of data using burst mode, which provides for the most efficient usage of the Micro Channel bandwidth.

A buffered adapter minimizes or eliminates the critical service time requirement of the adapter (see the section, "Critical Service Time").

Intelligent Adapters

An adapter can furnish the basic connectivity to a device, and it also can perform some of the tasks necessary for the device to operate. When an adapter furnishes more than just the basic connectivity, it requires some intelligence. This intelligence is generally in the form of an onboard processor. An onboard processor can perform functions that would normally be accomplished by the system processor. This offloading of function allows the system processor to perform concurrent execution for other tasks.

A programmable, intelligent adapter provides the user with the capability to develop applications that are executed on the adapter. The programmable adapter may be an I/O subsystem or a coprocessor. A coprocessor adapter may allow for system performance improvement of an existing system.

Adapter Functional Packaging

Single-Function Adapter: A single-function adapter is one that has a single Micro Channel interface. The adapter may have one or more operational modes. However, for an adapter with more than one operational mode, only one of these operational modes is active at any one time. The active operational mode determines the performance characteristics. A bus master SCSI

adapter is an example of a single-function adapter that supports a single operational mode. A parallel port adapter is an example of a single-function adapter that may support more than one operational mode. This adapter may act as an I/O slave as operational mode 1 and a DMA slave as operational mode 2.

An adapter can present a single Micro Channel interface, but support multiple I/O attachments. This is also a single-function adapter if the multiple attachments are managed transparently to the Micro Channel interface. An intelligent I/O subsystem that supports multiple serial ports and a SCSI adapter that attaches multiple devices on the SCSI interface are examples of this type of single-function adapter.

Multifunction Adapter: An adapter can contain multiple independent functions. The interface to the channel may have a common physical appearance, but there will be an independent logical appearance for each function. The functions may be identical with a single operational mode, such as a serial port adapter that supports two serial ports; or the functions may be different, such as an adapter that has a serial port and a parallel port. There is a unique operational mode for each function, in the second case. In both cases, each independent function can be considered a separate adapter, except for the physical resources. Therefore, the system resource requirements for the multifunction adapter are the total of the system resource requirements for each independent function.

Software

There are three classes of software that will support a given hardware

configuration. These software classes are application, operating system and device driver. Figure 10 shows the interrelationship among the software classes. It also shows the different levels of interfaces used to control the transfer of data to or from an adapter.

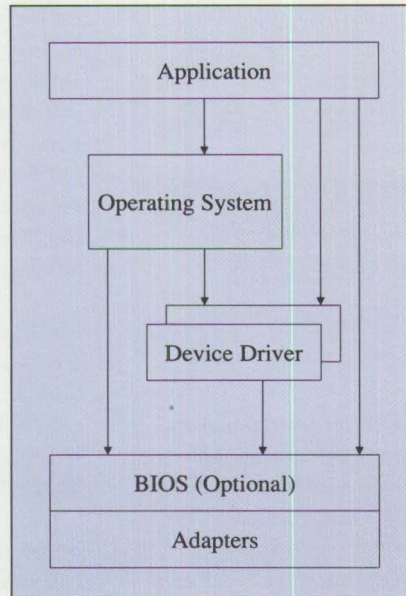


Figure 10. Software Systems Structure

When selecting the components for a system, both software and hardware, it is important to know that the best performance is realized when the software complements the hardware. In addition, the software, operating system, application and device driver should be matched for best performance. This section will address the aspects of software that affect system performance.

Applications

Application software performs a function for the user. Examples of applications are a spreadsheet, a financial model, or an engineering calculation.

Tasking Environment

When choosing an application, it is necessary to determine whether it is suitable for your environment. It is also necessary to determine the type of operating system that the application was designed to run under. Most applications that are designed to run under a single-tasking operating system will run under a multi-tasking operating system, but this operation may be inefficient. Such applications may monopolize resources at the expense of other tasks. However, applications specifically designed to run under a multi-tasking operating system in order to exploit the multitasking capabilities, use the resources in an efficient manner. This allows other tasks access to the resources.

Most applications fall into one of two basic categories, processor-intensive or I/O-intensive.

Processor Utilization

A processor-intensive application is one which spends at least 80 percent of its time performing calculations. Examples of this are engineering and scientific programs and financial modeling applications. Most business programs are only processor-intensive for short periods of time, for example, for the recalculation of a spreadsheet or the compiling of a program. The effect of a processor-intensive application may be observed as delays between a request and the response to the request.

I/O Utilization

An I/O-intensive application is one that spends at least 80 percent of its time performing I/O operations. Text editors, word processors or data base programs are examples of programs that are I/O-intensive. The throughput supported by an I/O-intensive program is determined by

its design. A program that performs character I/O is generally at the low end of the performance spectrum while a program that performs block I/O is generally at the high end. The size of the request blocks and the frequency of the requests will influence overall throughput.

Application Performance Consideration

The application response time, in a single-tasking environment, is determined by the hardware. In a multitasking environment, the resources required by the other tasks that are being executed concurrently can affect the application response time.

Operating Systems

The operating system provides system resource management and the support structure for executing application software and device drivers. The operating systems are general purpose and are designed to support a wide variety of application software.


Resource Management

The operating system is responsible for managing the system resources. This management involves the ownership and sharing of the resources. When the resources are examined, it is done relative to the environment, single-tasking or multitasking.

In a single-user, single-tasking environment, such as IBM DOS, the operating system is responsible for handling interrupts and I/O services. The I/O services handled by the operating system are console I/O and file I/O. All other I/O is performed by device drivers that are separate from the operating system. A single-user, single-tasking environment handles one task at a time.

A single-user, multitasking environment is more complex; IBM OS/2

is an example of a single-user, multitasking operating system. This environment allows a number of concurrent activities. The operating system is responsible for optimizing the overall system throughput by allowing the execution of multiple tasks based upon their relative priorities.



The operating system is responsible for managing the system resources.

Some of the services furnished by the operating system are the handling of interrupts, memory management, and task management. The operating system determines whether a given interrupt level can or cannot be shared and calls the appropriate interrupt handler.

The operating system will try to balance the throughput of system activities depending on task priority. This is accomplished through its task management services, the services used to switch from one task to another. Task switching can result in a call to the memory management services, which may result in I/O activity on the channel.

Multitasking operating systems rely on subsystems and device drivers to manage their respective I/O requirements. These subsystems and device drivers are responsible for handling I/O requests from the applications running in the system, and are also responsible for performance optimization. These subsystems can be acquired with the operating sys-

tem or they can be acquired as separate software products and installed as a part of the operating system.

A multiuser, multitasking operating system takes the complexity of the single-user, multitasking operating system and adds the support for multiple users, generally by supporting multiple terminals. This requires operating two or more terminal sessions concurrently. There are different versions of multiuser operating systems, and applications written for one may not necessarily run on another. IBM AIX is an example of a multiuser, multitasking operating system.

Multitasking operating systems that are executing I/O-intensive applications may fully utilize the resources available in a system. The user interface will reflect performance by its responsiveness to the requests.

Some application software packages can run a form of multitasking using a single-tasking operating system.

Device Drivers

Device drivers are written to handle the specific requirements of a particular device. They are designed to work with a specific operating system. There is at least one interface that the device driver uses to communicate with the operating system. Other interfaces may be provided to allow the device driver to communicate with programs running under the operating system. These interfaces allow the application software (above the device driver) to be device independent. The device driver makes it possible to support new devices after an operating system has been made available.

The device driver's characteristics and capabilities are dependent upon the device that is being supported and the environment that they are operating under.

Execution Environment

In a single-tasking environment, a device driver will support a device either through interrupts or through polling by the system processor. Events in a single-tasking environment are usually synchronous; for example, an application sends or requests data and then waits for an interrupt, or polls for the response. The device driver typically uses interrupts to support a device in a multitasking environment. It is expedient in this environment, because when one task is waiting for a request to be serviced another task can be executing instructions. Events in a multitasking environment are asynchronous, and the device driver must be able to support multiple requests. These device drivers use system resources more efficiently.

Devices Supported

A device driver can support a single device, or it can support multiple devices. The performance derived from a device driver may be a function of the capabilities of the device. If a device driver is supporting a character I/O device, it may buffer the requests coming or going to the application, but it still has to transfer one character at a time to the device. If a device driver is supporting a block I/O device, it will be able to transfer data a buffer at a time. The size of the buffer will depend on the capabilities of the device.

BIOS

The Basic Input/Output System (BIOS) was developed to provide a compatible interface between soft-

ware and hardware. BIOS allows the applications programmer to request an I/O operation without having to consider the physical details of hardware operation.

Two types of BIOS exist, BIOS and Advanced BIOS (ABIOS). BIOS is used in single-tasking environments such as IBM DOS. A BIOS is used in multitasking environments such as IBM OS/2.

Performance: Parameters and Considerations

The performance of a system is determined by a combination of the hardware capability, the operating environment and how it is used. Performance is a measure of system productivity.

It is necessary to determine the capacity that is needed to adequately perform the work required of the system. Then it is necessary to define a configuration that has the required capacity. To ensure that a system will meet its throughput and response time performance objectives, it is necessary to configure the components consistently with their capacity requirements. Since the channel provides the fundamental interconnection among the system processor and multiple adapters, the interaction of the system processor with the adapters becomes the most important part of the configuration process.

Performance Parameters

The parameters that define the hardware capability are system processor performance and the Micro Channel data transfer rate. The parameters that measure system perfor-

mance are throughput and response time.

System Processor Performance

The system processor performance is the measure of its ability to do work and is referred to as the *system processor power*. When viewed over the short-term, in the order of hundreds of microseconds, it is referred to as *instantaneous system processor power*; when viewed over the long-term, in the order of several seconds, it is called the *average system processor power*. In addition, in multitasking environments where there can be concurrent I/O, it is referred to as *effective system processor power*.

Instantaneous System Processor

Power: The instantaneous system processor power available in a system is determined by measurements taken over the short-term, several hundred microseconds at a maximum. It is dependent upon a number of factors. These factors include the operating environment, the system structure, and the adapters. In a single-tasking environment, the instantaneous system processor power available when the system processor is active is the rated power of the system processor. In a multitasking environment, the instantaneous system processor is affected by concurrent I/O and system structure.

The available instantaneous system processor power varies with time; this variability will be discussed under "Effective System Processor Power."

Average System Processor Power:

The average system processor power available in a system is determined by measurements taken over the long-term, several seconds or more. It is an indication of the over-

all system capability that is available to the user. It is not an indication of the system capability available to a user at any specific instant of time.

Effective System Processor Power:

The effective system processor power available to run applications is a function of the operating environment, the system structure and the amount of concurrent I/O. In a single-tasking environment, the system processor is either processing an application or waiting for an I/O operation to complete. When the system processor is processing an application, the rated power of the system processor is available. When the system processor is waiting for an I/O operation to complete, the system processor instruction execution is suspended. In a multitasking environment, two or more tasks can be active at the same time, and the system processor power can be affected by concurrent I/O requiring access to the channel.

Effective system processor power is the power available to an application after the effect of concurrent I/O has been taken into account. In the basic system structure as shown in Figure 2, the channel is used by the system processor to fetch instructions and data and by the I/O to transfer data. Therefore, the system processor's ability to fetch instructions or data from system memory can be delayed by I/O operations that share the channel. With the enhanced system structure, when the system processor fetches instructions or data from the memory cache, it uses the processor bus as shown in Figure 3. There is no contention with concurrent I/O, and the effective system processor power is the rated power of the system processor. This is one of the

reasons that the enhanced system structure can exhibit more system processor power than the basic system structure. When the system processor fetches instructions or data from system memory, it requires the memory bus/channel and its operation can be inhibited by I/O operations that share the channel.

The potential contention between the system processor and the I/O for use of the channel makes feature adapter selection an important consideration. In multitasking environments it becomes more important to select feature adapters that use the channel efficiently. These adapters use less Micro Channel time to transfer the same amount of data, and, therefore, reduce the potential for contention.

Micro Channel Bandwidth

The Micro Channel bandwidth is the data transfer capability of the channel. It is the throughput of the channel measured in MB/sec. When viewed over the short-term, this bandwidth is called *instantaneous bandwidth*, and when viewed over the long-term, it is called *average bandwidth*.

Instantaneous Bandwidth: When the system processor, or an adapter, uses the channel, the instantaneous bandwidth is of interest. Figure 9 shows the ranges of instantaneous bandwidth available for different adapter capabilities. When a participant is using the channel, the instantaneous bandwidth of the channel is based upon that participant's data transfer rate.

Average Bandwidth: The average bandwidth of the channel is the bandwidth that is delivered over a long period of time. When two, or more, participants require the Micro Channel, each participant is allo-

cated a portion of the bandwidth. (Micro Channel architecture permits 15 adapters, plus the system processor, to simultaneously request bandwidth.) Therefore, the average bandwidth is the summation, over time, of the active participants' instantaneous bandwidths.

Throughput

Throughput is a measure of the amount of work that a system can produce. It is the number of tasks that can be performed in a unit of time. Throughput is affected by the capabilities of the hardware, the application software and the operating system.

Response Time

Response time is the length of time required to service a request. It is measured from the time that the request is placed until the time that the response to that request is received.

Considerations

There are a number of performance-related considerations involved in selecting a system. These include the operating environment, the application software, the system structure and the adapters.

When the components of the system are matched, it will deliver a satisfactory level of performance. The performance may be affected by mismatched components. Therefore, care is necessary in matching the capabilities of the system hardware, the adapters and the software to each other and to the functions to be performed.

Some hardware configurations work better in a single-tasking environment. One such configuration has adapters that have a critical service time dependency on the system processor. These adapters depend

upon the resources being available when needed. When one of these configurations is used in a multitasking environment, there is the possibility that degraded performance, recoverable errors or operational faults might occur. This problem can be avoided by selecting adapters that do not have a critical service time dependency on the system processor. This configuration should increase the system capacity and improve the system performance.

A properly loaded system, with matched components, can exhibit degraded performance at specific instants of time. These systems will slow down when the instantaneous workload peaks and temporarily exceeds the system capacity. The system load stabilizes, and the performance returns to normal when the workload peak is reduced.

Critical Service Time

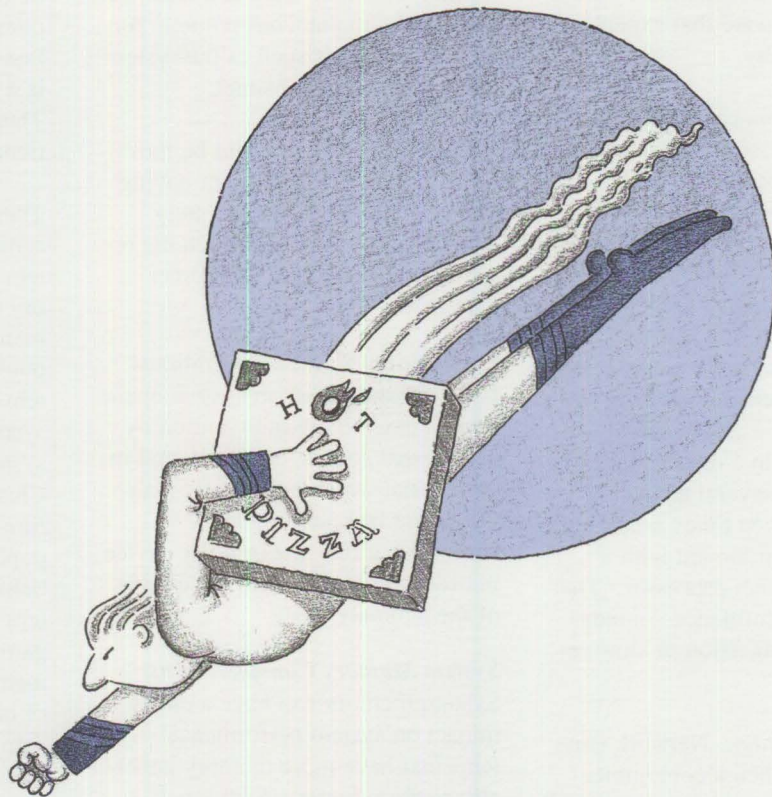
An adapter that does not arbitrate for the channel must be serviced by a controlling master. The controlling master can poll the adapter to initiate a data transfer, or the adapter can signal the controlling master with an interrupt to indicate it is ready for service. The time required for a controlling master to service an adapter is referred to as the *service time*. The amount of time that an adapter can wait for service is called the *critical service time*.

Micro Channel architecture describes the arbitration process and the maximum delay that a participant can experience when the channel is active. It is recommended that an adapter have a buffer large enough to overcome the arbitration delays. If an adapter does not have sufficient buffer capacity, there is the possibility of a *data overrun*. A

data overrun occurs when an adapter is receiving data from a source and exhausts its local buffering capability before it is serviced. The time interval before the adapter is serviced is the Micro Channel service time. If the adapter does not have sufficient buffer capacity, the time interval before this capacity is exhausted is the Micro Channel critical service time.

A *control overrun* occurs when an adapter does not receive control information in a timely fashion. This overrun can occur when a critical service time or a Micro Channel critical service time is not met.

When a critical service time is not met, an exception condition occurs, and the system will experience a performance degradation, a recoverable error or an operational fault.



Performance Degradation:

Performance degradation occurs when a device does not receive service within its critical service time period. The result of this service anomaly is that there is a perceived slowdown in system operation. User intervention is not required, and operations return to normal when the instantaneous power reaches a level that is adequate to handle the current workload.

Examples of this are character mode, line and buffered page printers. These printers stop operating until the next character, line or page is available, and then resume their operation.

Recoverable Errors: Recoverable errors are anomalies in system performance that do not require intervention from an external source.

These problems are handled by the hardware or software that experiences the difficulty.

Recoverable errors can occur when the system processor's instantaneous power is not sufficient to maintain the required level of system performance. For example, an adapter's critical service time is not met. This will usually occur when a system configuration is mismatched. Examples are an adapter with a short critical service time designed for use in a single-tasking environment, used in a multitasking environment, or several adapters with critical service times used in a multitasking environment with a low capacity system processor. The result can be performance degradation, network congestion or slow response time.

Network Congestion: Network congestion occurs when a communication facility does not receive service within a critical service time. The result of this service anomaly is a slowdown in network operations. This is caused by the network retrying the failed operation. User intervention is generally not required. Operation returns to normal when the instantaneous power reaches a level that is adequate to handle the current workload.

An example of this is a local area network (LAN) adapter where the buffers are full. The LAN adapter cannot receive any additional messages, and the LAN becomes clogged with messages for the saturated station. This situation continues until buffers on the saturated LAN adapter become available.

Slow Response Time: Slow response time is the result of the system processor not having sufficient capacity to service a user request in

a timely manner. This is a result of too many simultaneous requests for system resources such as the system processor and the channel.

An example of this would be moving a mouse device and not having the screen respond immediately. The request is satisfied when the required system resource becomes available.

Operational Faults: Operational faults are anomalies in system operation. They require intervention by an external source. This external intervention, which is typically manual, could be restarting a communications line or other device in order to recover from the effects of the anomaly.

System Memory Considerations

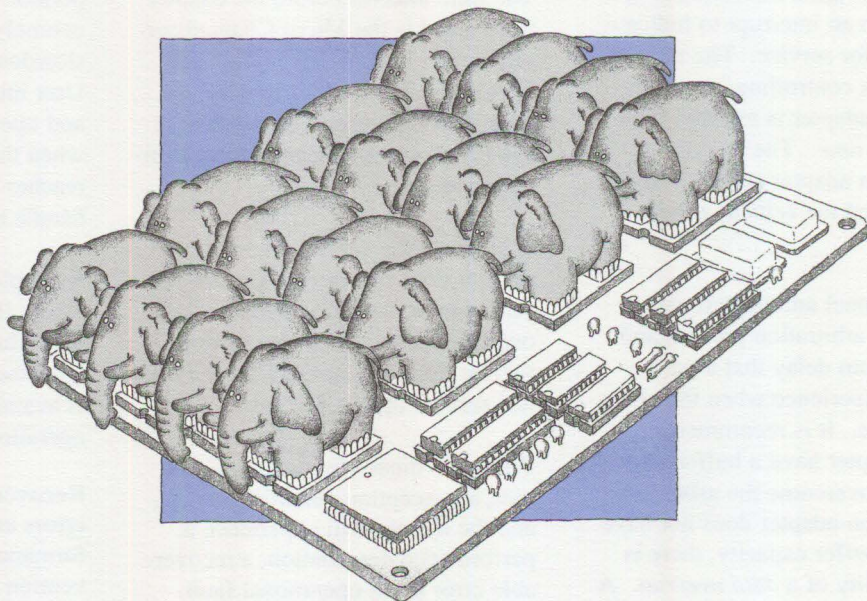
System memory can have a major impact on system performance. Systems that have more memory generally perform better. Additional memory reduces the need for paging, or swapping, information from a disk file, thereby reducing the delays incurred waiting for disk access. Also, additional memory can

be used as a disk cache, where frequently used disk data is stored for fast access, or a RAM disk, which is a form of fast access disk file. There are other similar uses for additional system memory.

These performance gains are achieved whether the additional system memory is system board memory or Micro Channel system memory. On some IBM PS/2 computer systems, the memory subsystem operation may provide the system processor a faster access to system board memory than to Micro Channel system memory. Therefore, it is advisable to completely populate the system board memory before adding Micro Channel system memory. The performance gains achieved by using additional memory, either on the system board or on the channel, are far greater than the performance differences that might exist between system board memory and Micro Channel system memory.

Address Bus Considerations

A 32-bit system processor with a 32-bit address bus has performance ad-



vantages over systems that support 24-bit addressing. The 32-bit address bus supports an address space of 4 GB. The 24-bit address bus supports an address space of 16 MB. When an adapter in a 32-bit operating system, (for example IBM AIX), has a 24-bit address bus, the software must manage the 16 MB data space to assure that the address for a data transfer between an adapter and system memory resides within the 16 MB space. This software overhead is eliminated, yielding better system performance, when the adapter has a 32-bit address bus.

Data Transfer Considerations

Data transfer can be affected by a number of factors. These factors include the data transfer time, the data transfer width, the adapter buffer and burst capability, and the number of physical moves required.

An adapter that can support the minimum default data transfer time of 200 nanoseconds will move data faster than an otherwise equivalent adapter that supports an extended data transfer time of 300 nanoseconds (see Figure 9). The wider data transfer (for example, 32 bits as opposed to 16 bits) will have a higher data transfer rate for the same data transfer time.

The Micro Channel arbitration time is a minimum of 300 nanoseconds. An adapter that supports burst mode can perform multiple data transfers after one arbitration. This can cut the arbitration overhead up to 97 percent for long bursts and greatly increase the Micro Channel data transfer efficiency.

The actual data transfer efficiency is affected by the number of moves required to get the data to the proper place. For example, to move data

from a disk file to a LAN adapter could use any of the adapter types. Because this type of data transfer typically involves large files, only the DMA slave adapters and the bus master adapters will be considered because they can support burst mode. An adapter that supports burst mode will provide a buffer to sustain the burst. The size of the buffer may affect the length of burst that the adapter can sustain. DMA slave adapters require four physical transfers to move the data from the disk file adapter to the LAN adapter. The following physical moves are required to complete the move from the disk file: first from the disk file adapter to the DMA controller and then from the DMA controller to system memory. The following physical moves are now required to complete the move to the LAN adapter: first from system memory to the DMA controller and then from the DMA controller to the LAN adapter. Bus master adapters will require two physical transfers to move the data from the disk file adapter to the LAN adapter: one move from the disk file adapter to system memory, and a second move from system memory to the LAN adapter. Bus master adapters that support adapter-to-adapter data transfer will require only one physical move to accomplish this transfer. This move would be from the disk file adapter directly to non-system memory on the LAN adapter.

The bus master adapter in the previous example has an advantage over the DMA slave adapter in addition to using fewer physical moves. The bus master adapter can support a 32-bit data transfer while the DMA slave adapter is currently limited to a 16-bit data transfer. Therefore, the bus master adapter has a greater channel efficiency than a DMA slave adapter; it requires fewer phys-

ical moves to complete a data transfer, and each physical move can contain twice the data.

Adapter Participant

The adapter participants reflect the Micro Channel participants, (see the section, *Micro Channel Participants*), and are either masters or slaves. Each participant has capabilities that make it suitable for use in different operating environments and for performing different tasks.

I/O Slave Adapter: The I/O slave adapter has a critical service time dependency on the master (system processor or the bus master). It generally does not provide a data buffer and requires a controlling master to perform data transfers.

The I/O slave adapter is most suitable for use in single-tasking environments. It may also be adequate for use in multitasking environments that have low data transfer rate requirements.

Memory Slave Adapter: A memory slave adapter may be a Micro Channel system memory adapter or a memory-mapped I/O adapter. A Micro Channel system memory adapter does not have a critical service time dependency on the master (system processor or the bus master) or the channel. A memory-mapped I/O adapter typically does not have a critical service time dependency on the master (system processor or the bus master) or the channel.

The memory-mapped I/O adapter, which provides memory for use as a data buffer, requires a controlling master to perform data transfers. The controlling master can be the DMA controller, a bus master or the system processor. If the controlling master is either the DMA controller

or a bus master, this adapter is suitable for use in any operating environment.

When the system processor is the controlling master, the system processor is required to execute instructions to perform each data transfer. Therefore, this adapter is most suitable for use in single-tasking environments and for multitasking environments where the data transfer requirements are low. The memory-mapped I/O adapter can use an excessive amount of the system processor resources when it is required to transfer large amounts of data. In a multiuser environment, this excessive use of the system processor resources can adversely affect system performance.

DMA Slave Adapter: A DMA slave adapter may be buffered or non-buffered. Typically, a buffered DMA slave adapter does not have a critical service time dependency on the master (system processor or the bus master) or the channel, and a non-buffered DMA slave adapter has a critical service time dependency on the master (system processor or the bus master) or the channel.

The DMA slave adapter relies on the DMA controller to be the controlling master to perform data transfers. This adapter requires two physical moves for each data transfer. There is one move between the adapter and the DMA controller and another move between the DMA controller and system memory. The DMA slave adapter supports burst mode data transfer, which allows multiple data transfers for each arbitration.

The DMA slave adapter is suitable for use in any environment. It is best used when the data transfer re-

quirements are low to moderate because of its Micro Channel efficiency.

Bus Master Adapter: A bus master adapter may be buffered or non-buffered. Typically, a buffered bus master adapter does not have a critical service time dependency on the master (system processor or the bus master) or the channel, and a non-buffered Bus Master adapter has a critical service time dependency on the master (system processor or the bus master) or the channel.

The bus master adapter is generally the most sophisticated Micro Channel participant. It is a controlling master and does not require any system processor intervention to perform data transfers. The Subsystem Control Block (SCB) architecture provides a software protocol for a bus master to transfer control information and data between bus master adapters and between a bus master adapter and the system processor. This results in a further reduction of system processor dependency on the part the bus master adapter.

The bus master adapter can move data very efficiently. It requires only one physical move for each data transfer. Burst mode is supported where multiple data transfers can be completed for each arbitration. In addition, bus master adapters can support adapter-to-adapter data transfer, which allows data to be transferred directly between two adapters without an intermediate transfer to system memory.

The bus master adapter is suitable for use in any environment. Its Micro Channel efficiency makes it desirable for use when large amounts of data are to be transferred in a multitasking environment. The bus master adapter is

recommended for use in all multitasking environments, and it is especially useful in high data rate demand environments.

Product Selection

The previous sections have discussed the various components of a Micro Channel computer system, and the performance parameters and considerations of those components. This section builds on that base and provides help in selecting and configuring Micro Channel systems and feature adapters.

Operating Environment

The most important consideration in selecting and configuring a hardware system is the intended usage, and, based on the intended usage, the operating environment. The operating environment can be either single-tasking or multitasking. A single-tasking environment is for a single user. A multitasking environment may be either for a single user or for multiple users. The software is selected based on the established operating environment and the intended usage.

Applications: Applications are selected that meet user requirements and are compatible with the operating environment. The applications must be selected in conjunction with the operating system to assure compatibility.

Operating System: The operating system is selected based upon the operating environment, and the available applications. For example, IBM DOS supports a single-tasking environment; IBM OS/2 supports a single-user, multitasking environment; and IBM AIX supports a multiuser, multitasking environment.

Device Driver: The device driver must be compatible with the operating system and the adapter. The device driver can be provided with the software, with the adapter or as an independent entity. If the device driver is provided with the software, the adapter hardware requirements are specified. If the device driver is not provided with the software, it must be provided with the adapter or purchased separately.

Hardware Selection

Hardware system selection consists of determining the hardware requirements of the selected software, the operating environment, the system capacity, and the usage environment, home or business.

Based on the above, the system hardware is selected. The considerations are the system performance, the amount of installed system board memory, the maximum capacity of system board memory, the I/O capability of the base system, the FCC classification and the resources available.

After the system hardware is selected, the optional features required are determined. These optional features include memory expansion, system board memory or Micro Channel system memory, and other feature adapters (for example, coprocessor, ESDI, SCSI, tape, serial port, display, printers, plotters, facsimile). The considerations in the feature adapter selection are the functional capability, the FCC classification, the performance attributes of the adapter and the resource requirements.

If the device driver is provided with the operating system or the application, it is necessary to select an adapter that is compatible with the

requirements specified by the software.

After the initial selection of the feature adapters, the total physical resources and system resources required by the system configuration must be reviewed. Any mismatches may require the selection of a different system, and/or feature adapters.

FCC Classification: The FCC has two classifications for computing devices. These are class A for commercial or industrial use and class B for residential or home use.

Physical Resources

There are physical resources that must be considered when a system is being configured. These resources include the maximum amount, or capacity, of system memory that can be supported, the amount of system board memory that can be accommodated, the number and type of Micro Channel connectors on the system board available for feature adapters, the number and type of Micro Channel connectors required by an adapter, and the number of adapters of the same type that can be coresident in a system.

System Memory Capacity: The hardware system specifies the maximum system memory capacity that may be accommodated. A system with a 24-bit address bus is limited to supporting a maximum of 16 MB of system memory. A system with a 32-bit address bus optionally may support greater than 16 MB of system memory, up to a maximum specified by the system. The system specifies the maximum system memory that may be accommodated on the system board. The expansion of system memory beyond that accommodated on the system mem-

ory board is accommodated by a Micro Channel system memory adapter.

Micro Channel Connectors: Each system board has a physical limit to the number of adapter cards that can be accommodated. This limit is specified in the number of connectors that are available. If a system board has three connectors available for options, then three feature adapter cards can be accommodated. Likewise, if a system board has seven connectors available for options, then seven feature adapter cards can be accommodated. When a greater number of feature adapter cards are required than the number of available connectors, either a system with additional Micro Channel connectors or a multifunction feature adapter card is required.

Micro Channel Connector Types: There are five types of Micro Channel connectors:

- Type 1 - 16-bit connector
- Type 2 - 16-bit connector with video extension
- Type 3 - 32-bit connector
- Type 4 - 32-bit connector with video extension
- Type 5 - 32-bit connector with matched-memory extension.

Based on the adapter connector type, Figure 11 indicates which of the five Micro Channel connector types that an adapter can use on the system board.

Note 1: A 16/32-bit adapter connector is an adapter with a 32-bit connector that is designed to allow

insertion into a 16-bit Micro Channel connector. In this case, the 32-bit adapter will operate as a 16-bit adapter.

Multiple Occurrences of an

Adapter: Multiple occurrences of an adapter is the maximum number of adapters of the same type (part number) that may be coresident in a system configuration. If configuration requirements cannot be met based upon the multiple occurrence capability of the adapter, it will be necessary to select a different adapter to provide the the required function.

System Resources

The Micro Channel system and adapters support a Programmable Option Select (POS) facility that eliminates the need for switches on the system board and adapters. An automatic configuration utility and a change configuration utility are provided with the system on the System Reference Diskette. The automatic configuration utility identifies the installed hardware and automatically tracks and allocates system resources (interrupt levels, arbitration levels, DMA channels, I/O address space - I/O ports, and memory address space). Information is used in the creation of the configuration data from the adapter

description file, the adapter description program, and, if required, an initialization program. The programs are provided with the system or the feature adapter. When more than one adapter is configured to the same system resource and that resource cannot be shared, only one of the conflicting adapters is enabled.

In special cases, it may be necessary for the user to change the default configuration settings from those set by automatic configuration. The change configuration utility supports a manual procedure that is used to resolve unusual conflicts or for performance tuning.

The above procedure simplifies the management and consideration of the system resources in adapter selection. However, it is advisable for the user to review the following system resource considerations in product selection to prevent a conflict, or to prevent exceeding the available system resources. Both the base adapter and the feature adapter requirements are included in the total system resource requirements for product selection.

DMA Channel: Each DMA slave adapter requires one or more DMA channels. The total of the base adapter and the feature adapter

DMA channels is limited to the total provided by the system DMA controller. If multiple base or feature adapters require a fixed assignment for a DMA channel, then it is necessary to check for potential conflicts in the selection of a DMA slave adapter.

Arbitration Level: Each DMA slave adapter requires one arbitration level per DMA channel assigned to the adapter. Each bus master adapter requires one or more arbitration levels. The total number of arbitration levels for the base and feature DMA slave and bus master adapters is limited to 15. If multiple base or feature adapters require a fixed assignment for an arbitration level, then it is necessary to check for potential conflicts in the selection of a DMA slave or a bus master feature adapter.

Memory Address Assignment:

A memory slave adapter, or an adapter that supports a memory slave participant, requires the assignment of one or more ranges of addresses within the Micro Channel memory address space. There are two types of memory slave participants that must be considered, Micro Channel system memory and non-system memory. Non-system memory is memory on an adapter

Adapter Connector	System Board Connector				
	Type 1	Type 2	Type 3	Type 4	Type 5
16-bit	X	X	X	X	X
16-bit with video extension		X		X	
32-bit			X	X	X
16/32-bit (see note 1)	X	X	X	X	X
32-bit with video extension				X	
32-bit with matched-memory extension					X

Figure 11. Micro Channel Connector Type

that is contained in the Micro Channel memory address space.

Micro Channel system memory:

The memory address range can be assigned between 1 MB and 16 MB where the system or the adapter limits the address bus to 24 bits. The memory address range can be assigned between 1 MB and 4 GB where both the system and the adapter have a 32-bit address bus. Automatic configuration assumes that the memory address is assignable on 1 MB boundaries. If the memory card does not support assignment on 1 MB boundaries, it may be necessary to perform manual configuration through the use of the change configuration utility. If the Micro Channel system memory adapter requires the support of an initialization program, a fixed disk is required in the system configuration.

Non-system memory: For an adapter that contains non-system memory, the memory address must be assigned in the adapter ROM address range (HEX 000C0000 to 000DFFFF), the address range between 1 MB and 16 MB, or the address range between 16 MB and 4 GB. The adapter specifies the number of memory addresses required for each memory address range that is supported.

If an adapter memory address requirement is restricted to the adapter ROM address range and the starting address is programmable on 8 KB boundaries, there is no addressing conflict. This scheme supports up to sixteen 8 KB segments without an addressing conflict. However, for an adapter that requires a fixed address range, it is necessary to check for potential conflicts of the memory addresses. If multiple adapters that are restricted

to the adapter ROM address range require greater than 8 KB of addresses each, then it is necessary to confirm that they do not exceed the 128 KB that is available. In addition, it is necessary to confirm that the address range can be configured in a contiguous block without an addressing conflict.

If an adapter supports a programmable range of addresses for non-system memory that is not constrained to the adapter ROM address space, there is no addressing conflict. The memory address can be assigned between 1 MB and 16 MB where the system or the adapter limits the address bus to 24 bits. The memory address range can be assigned between 1 MB and 4 GB where both the system and the adapter have a 32-bit address bus.

I/O Address Assignment: An I/O slave adapter, or an adapter that supports an I/O slave participant, requires the assignment of a block of one or more addresses within the Micro Channel I/O address space. The block of I/O addresses may be either fixed, a fixed base I/O address with a programmable Adapter I/O Address Select, or selectable by the Device I/O Address Assignment. If the adapter supports the Device I/O Address Assignment, up to 59 adapters of the same type can be supported in a system. An I/O address space conflict cannot occur for this case. If the adapter supports a fixed block of I/O addresses with a programmable Adapter I/O Address Select, up to eight adapters of the same type can be supported in a system. If the adapter supports a fixed block of I/O addresses, or a fixed block of I/O addresses with the programmable Adapter I/O Address Select, it is necessary to check for potential conflicts of the I/O address blocks between adapters.

Product Selection Summary

The proper selection and configuration of a system requires the matching of the system hardware and feature adapters to the selected software. The system hardware should have the performance and the capacity to execute the applications of interest. Information concerning IBM PS/2 computer system hardware specifications is in *Personal Systems Hardware Interface Technical Reference*, form number S68X-2330, available from IBM. This technical reference manual will also have information regarding the resources, both physical and system, available to each system.

The information of interest for system units includes:

- System board devices and features
- Performance specifications
- Physical specifications

Adapter information, performance and resources (both physical and system) required, should be obtained from the adapter manufacturer. A catalog of Micro Channel adapters, with manufacturers name, address, telephone number, and a brief product description is available from IBM. This catalog is *Catalog of Micro Channel Hardware Products*, form number G360-2824.

The information of interest for adapters includes:

- Adapter participant type
- Micro Channel connectors, number and types required

- DMA channels, number and assignment
- Arbitration levels, number and assignment
- Memory addresses, number and assignments
- I/O addresses, number and assignments
- Address bus width
- Data bus width
- Data transfer type and time
- Buffer capacity
- Burst mode support
- FCC classification

Refer to the article, "Information for Developers," in this issue for more information.

It is necessary to determine the resource requirements for each adapter after selecting the feature adapters of interest. The resource requirements are then summed by subcategory within the physical and system categories. The configuration is valid if none of the system hardware resources has been exceeded and there are no conflicts for fixed resource requirements.

If a resource has been exceeded, or there is a conflict for a fixed resource requirement, it is necessary to make new adapter selections and reassess the resources required, as

above. If a valid configuration is not obtained, it is necessary to make new adapter selections or to select system hardware with greater resources.

Reconfiguration should not be necessary under most circumstances. Typically, only a system configured for heavy-duty, multitasking operation will exceed system resources. It is more probable that a configuration problem can occur when an adapter requires a fixed resource.

ABOUT THE AUTHORS

Carl H. Grant is a senior technical staff member at IBM's Entry System Division in Boca Raton, Florida and is currently with PS/2 Systems Architecture. He is involved in the development of Micro Channel architecture. Carl joined IBM in 1960 and has previously been associated with development of communications products and architecture, IBM System/360 and 370 Channel and I/O development, IBM Distributed Systems early definition of IBM Token-Ring, and 9370 and AS400 communications IOP development. Carl has an Outstanding Contribution Award for the 2880 Channel, 2305, 3330 Integration and an Outstanding Innovation Award for 8100 Direct Attach Loop. He has two patents and several invention disclosures published in the "IBM Technical Disclosure Bulletin." He has a bachelor of science degree in electrical engineering from the University of Vermont.

Donald Ingerman is an advisory engineer in IBM's Entry Systems Division in Boca Raton, Florida.

He joined IBM in 1984 and is a member of the PS/2 Systems Performance Evaluation Department. Before joining the Entry Systems Division, Don did performance work on the SPD I/O Bus and several communications projects. He has a bachelor of science degree in electrical engineering from Fairleigh Dickinson University and a master of science degree in operations research from New York University. Don has been in the performance field for over 20 years with Norden Systems, Bell Telephone Laboratories, ARINC Research Corp, and the Library of Congress. He was a communications engineer with the Western Electric Company prior to his work in the performance field. His current responsibilities include PS/2 system performance and Micro Channel system performance.

Robert F. G. Robinson is a staff programmer at IBM's Entry System Division in Boca Raton, Florida and is currently with the OS/2 Market Planning group. He teaches classes and seminars on OS/2 and conducts customer briefings and disclosures on OS/2 Standard Edition. Robert joined IBM in 1974 and has previously designed and developed 3270 communications products for the IBM PC and minicomputers. He is coauthor of a book, OS/2 Presentation Manager Programming, to be published by Wiley and Sons in November, 1989. He has a bachelor of science degree from Florida Atlantic University.

Features and Benefits

*Les McDermott
IBM Corporation
Boca Raton, Florida*

This article discusses Micro Channel architecture, how it works, and the opportunity its use provides – to both end users and developers. The rationale for developing the architecture and the value it brings to modern computing systems is explained. Some key features of Micro Channel architecture are discussed.

IBM Micro Channel architecture is an information-transfer medium. Its primary purpose is to provide a very flexible, high-performance structure for moving instructions and data within a computer system. The term "performance" is used in this article to refer to the amount of "real work" done per unit of time. Real work refers to the actual output desired by the end user.

Micro Channel architecture, when implemented as a feature of a modern computing system, makes it possible for end users to purchase only the functions and performance needed today while ensuring that their future requirements can be met through enhancements and upgrades. This means that the customer's investment is for today as well as for tomorrow.

System Design Flexibility

IBM Micro Channel architecture enables today's developers to build systems with a wide range of function and performance without limiting their ability to expand those systems far beyond current implementations. The architecture allows end users full control of their envi-

ronment and, at the same time, satisfies their computing requirements as they change and grow over time.

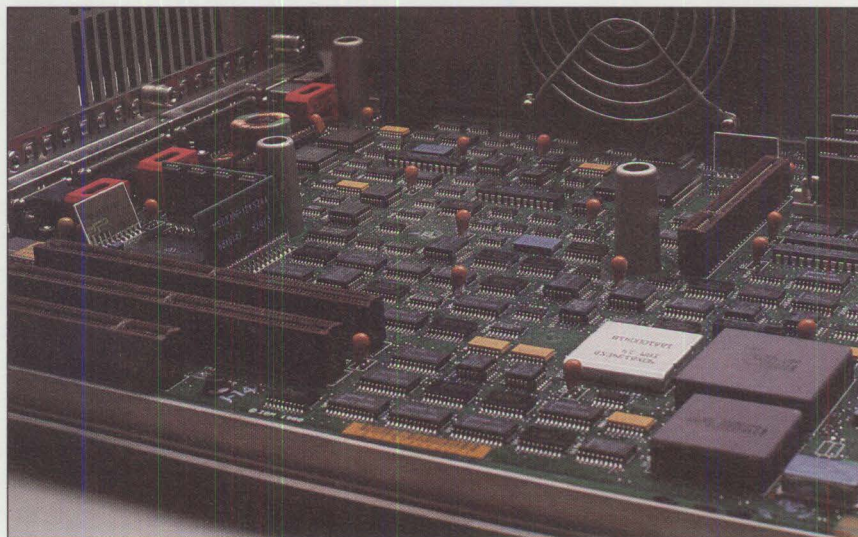
The IBM Personal System/2 family of products and any other modern computer systems that use Micro Channel architecture, unlike computer systems based on the AT bus and its derivative architectures, are not limited by the processor technology used to implement the system.

Micro Channel architecture allows processor technology to be isolated from the information-transfer medium by providing clear definition of the bus-master interface. Even the processor built into PS/2 systems is considered a bus master (in technical terms, the "default bus master"). This means that now multiple, equal (peer) bus masters based on highly dissimilar technologies can coexist in the system. The benefit to end users is that now they can enhance or upgrade their current systems to use the leading-edge technologies as they evolve. End users now have the flexibility to determine whether to enhance, upgrade, or replace their systems to meet their needs.

From Whence We Came

Before discussing the technical characteristics of the Micro Channel architecture, it is important to understand something of the history of personal computer bus architecture. The initial PC architecture was never intended to do what people have done with it. It was intended to be a single-user, stand-alone base – with relatively little use of communications – for home and small-business users. However, the IBM PC ended up being used to perform tasks far beyond our expectations. As a result, IBM designed an architecture for products that will not only meet end users' current needs, but can also be enhanced and upgraded to meet future needs – without adding costs to the initial purchase.

Requirements included workstation-environment flexibility, application portability, networking, host communications, and hardware compatibility across a broad range of computing environments. What was needed was a platform that would support the development of compatible systems from the low end of the product line – the single-user workstation – to the high-end,



host system environment. The platform would provide a new set of standards and requirements for today's small-systems computing environment.

To satisfy these requirements, and at the same time maximize design flexibility, three major architectures were evaluated: the software architecture composed of the operating system and applications; the logic technologies used to implement the hardware; and the information-transfer medium (bus/channel) used to move instructions and data. Making the information-transfer medium the platform on which to build provided the basis for allowing the most independence and flexibility.


An evaluation of the then-current AT architecture revealed inherent problems that would have to be circumvented. In the AT architecture, all three components – the software, the processor, and the information-transfer medium – were intimately tied together. What this meant was that when a system was announced, all of its functional capabilities were defined and the limits were permanently set. Instead of choosing to perpetuate these problems, IBM decided to design an architecture that would allow developers to provide compatible and portable products with extendable life expectancy.

This is a key reason behind our decision to develop the Micro Channel architecture. Our purpose was to eliminate extraneous requirements and remove the inherent limits on system flexibility.

A Technology for Today and Tomorrow

As an information-transfer medium, Micro Channel architecture is designed to allow end users to take ad-

vantage of emerging technologies. The architecture encourages the development of both high-performance and low-performance adapters on the same platform. This ensures that every adapter made, whether in 1987 or beyond the year 2000, can fit, function, and coexist with every other adapter that will be designed to Micro Channel architecture specifications.



Micro Channel architecture was defined to support multitasking and multiprocessing.

Micro Channel architecture was defined to support multitasking and multiprocessing environments that require the ability to move large amounts of data. The architecture removes the requirement on any system to drive the transfer of data and instructions at the slow rate at which instructions are executed in a microcomputer. By allowing data to be moved faster than it can be moved using instruction streams, you can substantially decrease the time needed to move information. This significantly increases the performance of the system without increasing the system-processor clock rate (the rate at which instructions are executed).

Micro Channel Features

Let's discuss Micro Channel architecture's primary features and the benefits they provide.

Like the foundation of a building, Micro Channel is a foundation – a support platform. In modern computing systems, it is the platform that provides the value, flexibility and performance capabilities. A superior information-transfer platform allows instructions or data to be moved from one place in the system to another without constraining the system's performance potential. A set of signals, the relational timings for these signals, and the procedures for using these signals is a fundamental requirement. A key benefit is the ability to provide existing systems with greater levels of functionality and performance through the definition of new procedures using the same signals and timings. Implementations of Micro Channel architecture can provide these platforms.

Wider Data Paths: The Micro Channel architecture provides a 32-bit address bus, a 32-bit data bus, a subchannel for audio, an arbitration sub-bus (for the ability to switch between multiple peer bus masters), and control signals. To maximize flexibility and minimize system implementation costs, a subset of the full Micro Channel is defined in the architecture. This system implementation subset has a 24-bit address bus and a 16-bit data bus. The full Micro Channel implementation is often referred to as the "32-bit Micro Channel." The defined system implementation subset is often referred to as the "16-bit Micro Channel." Plug-in adapter cards may be designed to the 32- or 16-bit implementation. The adapter-card developer is permitted to implement 8-, 16-, or 32-bit data paths.

Potential for System Growth:

A basic principle in Micro Channel architecture is that all implementa-

tions must be upward-compatible. That means all 16-bit implementations must be compatible with 32-bit implementations. It also means that all future implementations must be compatible with existing specifications.

Because multiple data path sizes are supported, the architecture requires all bus master and slave hardware implementations to be self-aligning on a cycle-by-cycle basis. This means that 32-bit adapter cards can be easily designed to plug into 16-bit systems and function with a 16-bit data path. Because all hardware implementations must be self-aligning and self-pacing on a cycle-by-cycle basis, software developers and end users do not have to be concerned about the physical machines or the physical structures. They do not have to be concerned about the compatibility of future offerings.

This is a significant benefit of owning computing systems based on Micro Channel architecture. As in the mainframe computer environment, today's small computer systems environment is subject to change. These changes place new requirements on the end-user systems. The defined flexibility, compatibility, self-aligning, self-pacing, and high-performance capabilities in every system that uses Micro Channel architecture permit new requirements to be satisfied using existing systems. End users are not required to purchase new special-purpose systems.

The system is not constrained by limitations of existing technology and physical phenomena caused by increasing information-transfer rates. For example, as you increase the system processor's clock rate, you also increase the system's susceptibility to internally generated

noise. In most other data-transfer architectures, this phenomenon either defines the performance limits of the architecture or increases the system costs.

One of the most common misconceptions about systems based on Micro Channel architecture is that they are more expensive than comparable systems using other information-transfer architectures. Anyone who chooses to make an objective investigation will find that Micro Channel-based systems cost less or approximately the same as other systems with equivalent function, quality and performance. Also, reduced service and support costs may significantly reduce the real cost of ownership.

Micro Channel-based systems can be enhanced and upgraded to new levels of performance and functionality.

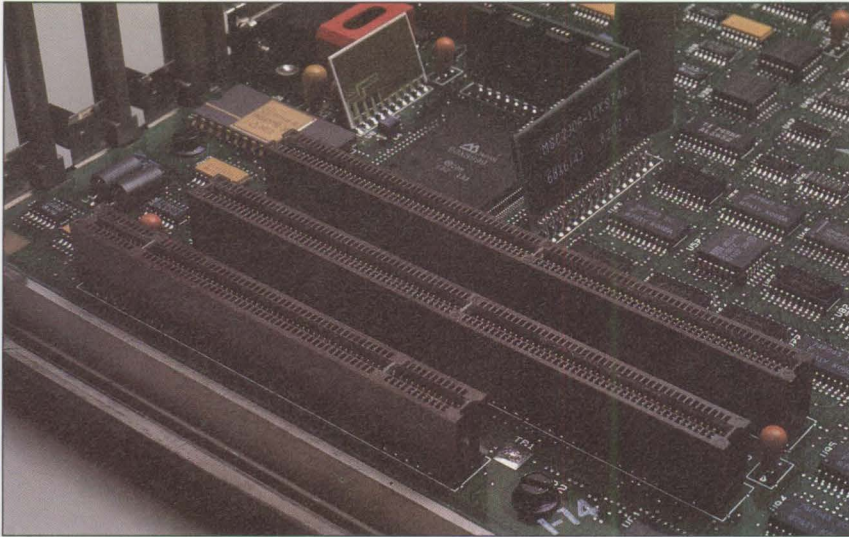
Information-Transfer-Rate Speed and Flexibility: The key characteristic of Micro Channel architecture is its ability to move information. In order to move it fast, the procedures must be simple and straightforward. Micro Channel architecture is a very straightforward structure. It specifies a set of procedures for the use of signals that are very well-defined and documented.

Transfer-rate flexibility is another important feature. Micro Channel

architecture provides a structure that defines rules for checking and handling while increasing the ability to move data and instructions at any particular period in time. Note that in this discussion we are referring to the ability to move information at a default (not a maximum) rate of 20 megabytes per second. While on a cycle-by-cycle basis, the bus master that owns the bus/channel (current bus owner) can decrease that rate to whatever is required by the hardware implementation. The cycle-by-cycle, self-pacing, and self-aligning features, along with the ability to define totally compatible higher-transfer-rate procedures, allow the default transfer rate to be increased. This can be done in a manner totally transparent to the application software.

As stated earlier, existing Micro Channel-based systems can be enhanced and upgraded to new levels of performance and functionality. These enhancements and upgrades are transparent to application software that does not directly manipulate the hardware. Only the physical device drive need change. The hardware determines precisely what the information transfers will be and what the transfer rates will be. To do this, signals have been grouped and timings specified to ensure that no implementation will adversely affect system functionality. On a cycle-by-cycle basis, this provides the maximum performance that any logic family, today or tomorrow, will provide to the end user.

Greater Configurability: Programmable Option Select (POS) is another feature of Micro Channel architecture. The function of POS extends far beyond the elimination of option switches. POS provides full control of the configuration,



functionality, and performance of the system. The configuration and functionality of hardware is controlled by the settings in the POS registers. For the first time in small systems, you have the ability to disable, diagnose, enhance, and reconfigure system resources under software control. The system software now has full control of the hardware and can determine the hardware functionality. System resources can be added or manipulated and dynamically reconfigured under software control. This provides enhanced opportunity in multitasking and multiprocessing environments such as those of OS/2 and IBM Advanced Interactive Executive (AIX). Through the use of Programmable Option Select, operating system software has the ability to control the system configuration, take system inventory using the software, and use NetView™ PC to control and diagnose the system.

Reduced Requirements for Interrupt Servicing: In addition to greater configurability, improved information-transfer capability and potential for system-growth, Micro Channel architecture provides a de-

sign that can reduce system overhead functions and code paths.

Micro Channel architecture can reduce the required hardware and software necessary to manage interrupt servicing. While fully compatible with existing code, new interrupt-handler code can significantly reduce the time spent servicing interrupts and maximize the time spent satisfying end-user requirements.

Interrupts are one of the most significant burdens on any software system. As more interrupt service is requested, more time and system-processor cycles are used to support system overhead. Micro Channel architecture's use of level-sensitive interrupts minimizes the number of interrupts that must be serviced as well as susceptibility to errors, spinning in interrupt loops and servicing false interrupts.

What is the significance of false interrupts? They decrease system performance. This performance loss depends on the physical environment and the code being executed. It turns out that in an environment for which IBM PC and AT architec-

tures were designed, increasing the transfer rates also increased the possibility of false, edge-triggered interrupt signals being generated by electromagnetic fields. False interrupts increase the frequency at which interrupt-correction routines must be executed, which, in turn, increases the time spent in servicing system-support software and reduces the functionality and performance of the system.

The number of interrupts requiring service can also be reduced through the use of bus masters. A bus master can be defined as a device that can request control of the bus/channel and potentially drive the address, the data, and the control buses. A bus master is not required to have a processor. Unlike bus masters in other architectures, a Micro Channel bus master can be implemented as a true peer to another bus master in the system.

A general-purpose bus master augments or upgrades the processing power of the system. An enhancement bus master improves system performance by implementing functions in a manner that reduces the software code execution typically required by a general-purpose bus master. A primary feature of enhancement bus masters is the offloading of the software's requirements, which are typically executed by a primary or alternate system processor and support logic. Enhancement bus masters provide logic specifically designed to implement functions in the most efficient manner on the transfer medium.

Bus masters also can reduce the need for multiple interrupts while implementing a function. Only one interrupt need be generated to indicate that a function has been completed.

Once again, new software can take advantage of this opportunity to increase system performance while being fully compatible with existing software. For application programs that are designed to function in older systems as well as in Micro Channel-based systems, IBM has provided a BIOS function (software interrupt 15 call) to identify Micro Channel-based systems. When a Micro Channel-based system is identified, the application software can bypass much of its interrupt-service code, thereby increasing the application's performance.

Telephone-Grade Audio: A feature allowed by the architecture is the audio sum node, which provides the ability to have true telephone-grade audio. An analog audio signal and an analog ground signal are specified. The bandwidth of the audio signal is 50 Hz to 10 KHz. The subsystem provides high-quality audio capability between devices and over the system speaker.

This subsystem eliminates the need to duplicate audio circuitry on devices added to the system. The audio sum node provides a new level of support for today's imaging and audio applications. It also provides a new level of opportunity for text-to-speech, voice-recognition, and full-motion audio / video applications.

A Superior Foundation

Micro Channel architecture provides a superior foundation for modern computing systems. It is a synchronous architecture chosen to maximize the life and flexibility of the architecture and systems that use it. The structure was also cho-

sen to minimize dependence on any processor or logic technologies.

Computer systems using this architecture have a well-defined transfer medium. This medium permits a mixture of attachment devices that operate at a wide variety of speeds.

The specification of the bus/channel also enables computer systems' function and performance to be upgraded and enhanced. Systems can be enhanced with attachment cards that contain additional function or provide improved performance. They can be upgraded by adding attachment cards that can control the system resources and increase the processing power of the system. Many years of effort were expended to ensure the flexibility, technology-independence and usability of Micro Channel architecture.

Micro Channel architecture is a strategically significant innovation in the computer industry. Like the IBM 360/370 I/O channel architecture, it was developed to be used in a wide variety and range of products. During development of this architecture, a large number of computer bus and channel architectures were evaluated. The best features and functions were identified, and specifications were created and structured to enable systems developers to provide the necessary set of functions without preventing others from being provided on attachment cards. The specification is written in a manner that permits end users to purchase the features and functions needed today while enabling them to add others as their requirements change.

ABOUT THE AUTHOR

Leslie F. McDermott is lead architect of Product Attachment Architecture, responsible for leading the development of the IBM low-end workstation attachment architecture. This includes promoting the implementation of the architectures in future workstation product development in all IBM divisions. Previously, he was with Entry Systems Related Products, responsible for providing technical information and assistance to internal IBM groups developing products related to the IBM Personal Computers. Other positions Les has had since joining IBM in 1976 include research in electromagnetic compatibility, development of data security techniques for communications and banking products, and design of manufacturing and test equipment. Les has a bachelor of science degree in electrical engineering from the University of Michigan. He is a graduate of the Sperry Vickers School of Hydraulic Power Systems for Industrial Machinery and of the IBM Systems Research Institute.

Bus Masters and Applications

*Patsy A. Bowlds, Ph.D.
IBM Corporation
Boca Raton, Florida*

This article provides an explanation of the bus master concept and defines terms used in discussing bus masters. It gives a historical perspective of the derivation of bus masters and discusses some of the benefits and applications.

The concept of bus masters is relatively new to the world of personal computers. IBM's introduction of bus master capabilities in Micro Channel systems in 1987 marked the beginnings of a new era in processing capabilities and applications in personal computers.

Some of these new applications and developers' tools and techniques are described in the subsequent articles in this publication. But before we

proceed, let's explore the bus master concept and its history, learn some useful definitions, and examine some of the benefits of bus masters.

What is a Bus Master?

A bus master is a function that allows an adapter card to gain control of the I/O bus and transfer information to or from either system memory or other adapters. The primary advantages of a bus master include increased throughput, design flexibility and greater processing power.

History

The evolution of the bus master concept goes back to some of the earliest days of IBM's computers. Figure 1 illustrates this evolution.

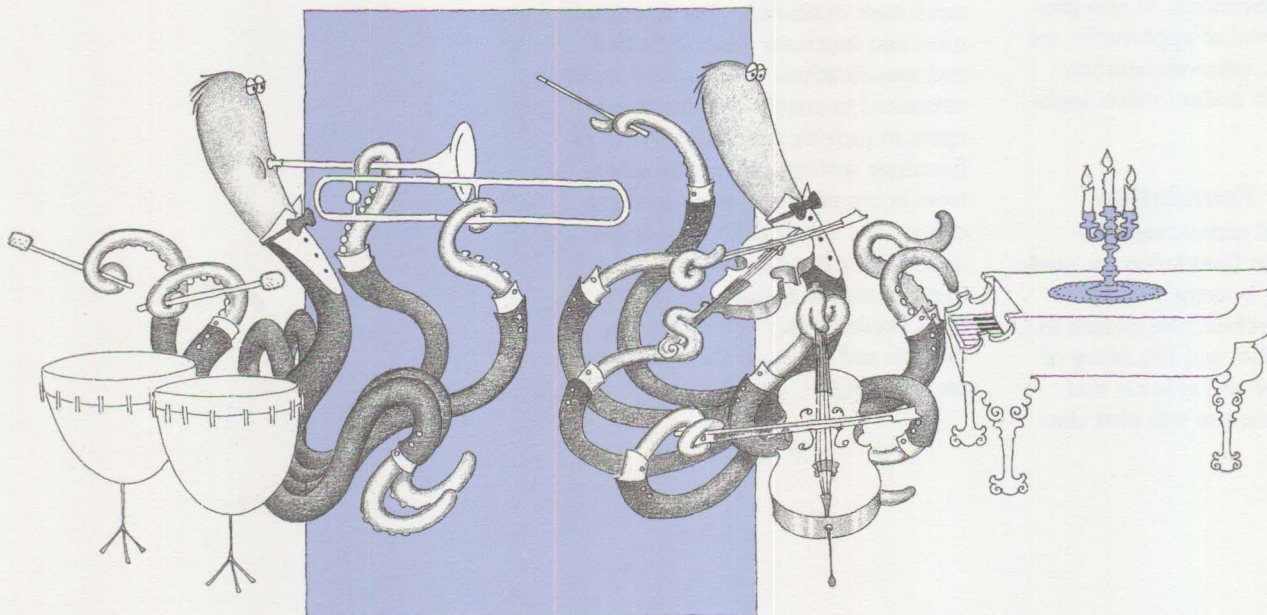
In the 1401 computer, all I/O was controlled directly by the CPU. With the introduction of the System/360 in 1964, I/O management was transferred from the CPU to the I/O processor. The CPU issued high-level commands (for instance, START I/O) to the I/O processor that subsequently managed all the

I/O, including data transfers to and from memory.

With the introduction of the System/7 in the early 1970s, I/O adapters were given direct access to system memory for the first time.

With the Series/1 architecture, came the concept of "smart I/O" or intelligent adapters. These adapters had direct memory access and most contained a microprocessor on board. Also added were new protocols that enabled high-level commands used in information transfers (control blocks).

The beginnings of a bus master concept in microcomputers first appeared in the Personal Computer AT in 1984. The AT was designed to support the operation of a single adapter with direct memory access (DMA) using a dedicated DMA channel. But because there was no provision in the architecture for the transfer of information directly to or from other adapters (peer-to-peer transfers), this was not a true bus master.



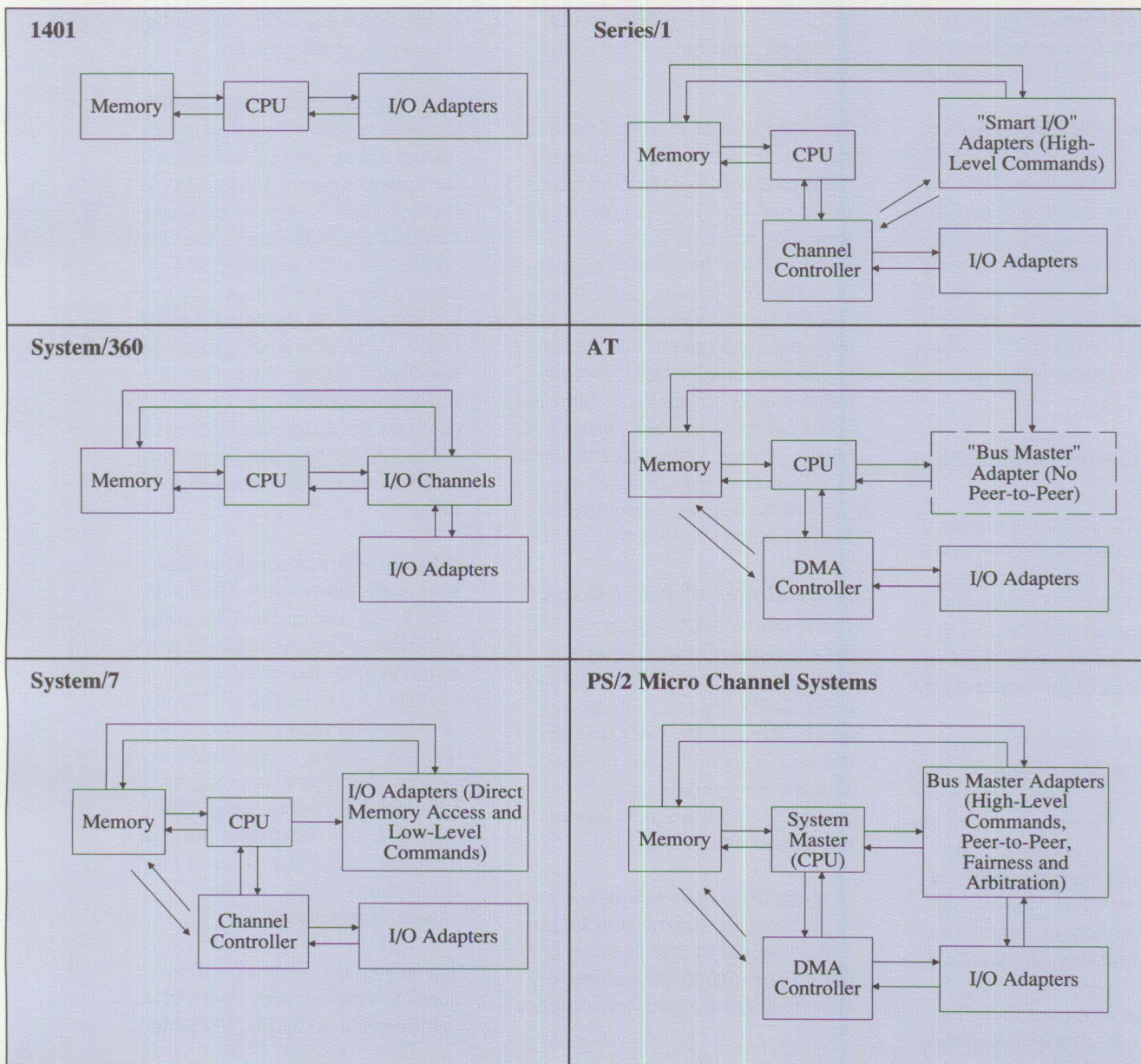


Figure 1. Evolution of the Bus Master Concept

The absence of arbitration and pre-emption capabilities in the AT architecture prevented the effective use of the bus by more than one "bus master." Without a defined method of equitable sharing of the bus, it would have been possible for a "bus master" to obtain control of the bus and prevent its use by other devices. One possible consequence of this

lack of fairness could have been the loss of memory refresh cycles that could have compromised data integrity. The protocols and procedures that could have prevented this and other problems were not part of the architecture. As a result, hardware developers were reluctant to use the feature for general-purpose applications. However, this design point

was quite appropriate for the way in which microcomputers were used in that time frame.

With the introduction of Micro Channel architecture in 1987, the basic elements and protocols that enabled true bus master capabilities were fully defined. Micro Channel architecture has an arbitration bus, a

method for preemption, and a fairness algorithm for equitable sharing of the bus by up to 15 bus masters.

The Programmable Option Select (POS) function in Micro Channel systems provides a method of configuring system resources that bus masters use (for example, arbitration levels) to avoid conflicts with other devices. The result of this design flexibility is that bus masters can be made to work in all systems, regardless of what is installed in them.

Micro Channel Information Transfers

In order to appreciate the virtues of bus masters and to understand how they work, let's examine the process of transferring I/O information in Micro Channel systems and the roles of some of the other types of masters defined in the architecture.

A bus master is one of three types of masters that Micro Channel architecture describes. All masters have knowledge of the origin and the destination of the information to be transferred and have the ability to manage the transfers. Slave adapters require the help of a master in information transfers (Figure 2).

The first type of master is the system master. It has the capabilities of assigning addresses and managing information transfers, but it also has the added duties of managing the system configuration. It may be the master that owns the Micro Channel bus whenever no other master has it (default master).

The second type of master is the DMA controller. This master manages the transfer of blocks of information to

and from adapters (DMA slaves) that require the help of the DMA controller.

The third type of master is the bus master. A bus master functionally replaces the role of the DMA controller and the system master in information transfers if the information to be transferred is to or from that bus master. This means that the system master issues a high-level command to the bus master, and the bus master takes over the transfer tasks. The DMA controller is not involved in a bus master's activities. Figure 3 illustrates the types of information transfers and the roles of the various "participants" in Micro Channel architecture.

Advantages of Bus Masters

The overwhelming significance of the capabilities of bus masters is that many of the functions that have traditionally depended on the system master and the DMA controller can now be done by bus masters. This design translates into decreased bus utilization, enhanced throughput and greater concurrency.

In terms of bus utilization, it is easy to see from the diagrams in Figure 3 that bus master transfers use one-half to one-quarter the number of steps required by the system master

or the DMA controller, which both involve three participants.

Bus masters can also significantly increase system throughput by reducing the number of interrupts to the system master. Reduction of this overhead is even more important than the reduction in bus utilization because the overhead and service time for interrupts are usually greater than the actual transfer time. These efficiency gains, combined with certain system latency times, for example, can be so significant that the difference between 16-bit and 32-bit bus master adapters may be minimized for some operations.

A bus master adapter often may have a microprocessor that can be used for executing complex commands to reduce some of the work of the system master, making it available for other tasks. The effect is to increase the number of tasks that can be done concurrently, a capability that is very appropriate for a multitasking operating system such as OS/2. Intelligent bus master adapters can also be used for distributed processing and for system-master upgrades.

The advantages of bus masters apply to systems with 16-bit I/O capabilities (for example, PS/2 Mod-

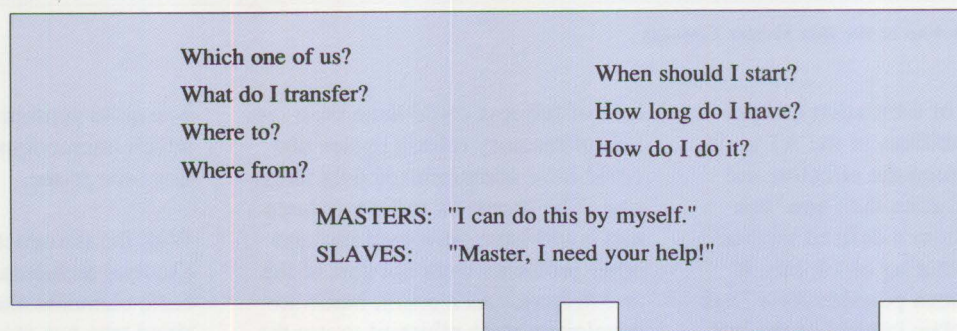
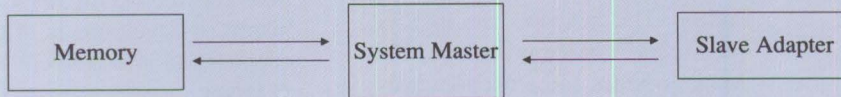
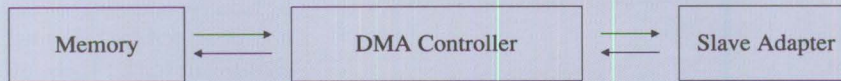


Figure 2. Information Transfers

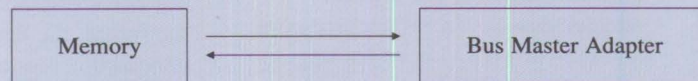
Type 1: Transfer to or from memory, to or from a slave adapter, via system master



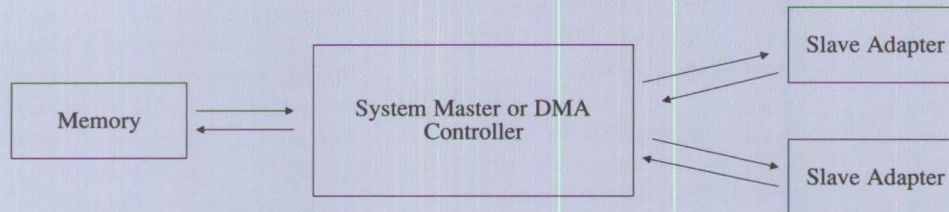
Type 2: Transfer to or from memory, to or from a slave adapter, via DMA controller (system master free to do other tasks)



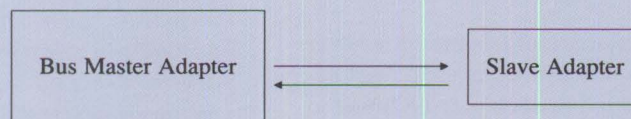
Type 3: Transfer to or from memory, to or from a bus master adapter (system master freed up further because of high-level commands)



Type 4: Transfer from one slave adapter to another (this is a combination of Types 1 and 2)



Type 5: Transfer to or from a slave adapter, to or from a bus master adapter



Type 6: Transfer from one bus master to another (Type 6 is a special case of Type 5 because one of the bus master adapters is being used like a slave adapter)

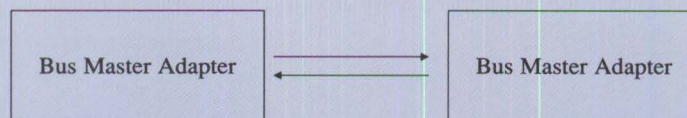


Figure 3. Micro Channel Information Transfers

els 50 Z and 55 SX), as well as to those with a 32-bit I/O bus, thus providing benefits to a broad range of users.

Bus Master Applications

Let's look at some of the exciting applications for bus masters.

Bus master applications fall primarily into two categories: those that enhance system performance or add new function, and those that provide design flexibility. These attributes can translate into investment protection and the ability to adapt to changes in how systems are used. Some of these applications are described in depth later in this publication.

Let's start with applications that enhance performance. One example of this type of bus master is local area network (LAN) adapters. Both Token-Ring and Ethernet bus master adapters are currently available for Micro Channel systems. Direct memory access and peer-to-peer transfers can significantly enhance throughput in these applications.

Other currently available applications are bus masters with intelligent controllers that manage extensive subsystems – graphics, image, and storage applications. Both intelligent ESDI and SCSI adapters (enhanced storage device interface and small computer systems interface, respectively) are available for Micro Channel systems. These adapters are especially effective because the intelligence on the adapter card, instead of the system master, can be used to manage functions of the storage devices. SCSI bus master adapters are described later in this issue.

An outstanding example of a bus master that introduces new function, boosts performance and provides design flexibility is the PS/2 Wizard adapter, also described in this publication. This adapter, with its RISC-based i860™ microprocessor from Intel, provides some phenomenal benefits as an application accelerator. It also provides the ability to run RISC (reduced instruction set computer) applications – an example of the flexibility that bus masters offer.

Another application that demonstrates the design flexibility is the use of a bus master to upgrade the system master. A currently available example is a bus master adapter that has an 80386 processor and onboard RAM used for upgrading an 80286 system, such as the PS/2 Model 60. This same capability could be used to upgrade an 80386 system to an 80486 system in the future. This investment protection is a built-in feature of every Micro Channel system.

As you can see from the previous examples, bus masters can greatly increase the processing power of systems through the use of multiple processors, and will open up many exciting applications in the future.

A summary of bus master adapters currently available and a brief description of each may be found in the *Catalog of Micro Channel Hardware Products* (G360-2824), available through your nearest IBM branch office or IBM Authorized Dealer, or by dialing the IBM PC User Group Support Bulletin Board at (404) 988-2790.

Summary

The introduction of bus masters into the Personal System/2 heralds a new era in cost-effective applications for microcomputers. IBM's historical development of the concepts of I/O management inherent in larger systems provided the basis for integrating bus masters in Micro Channel systems.

The performance advantages, design flexibility, and investment protection provided by bus masters can be significant for all types of users. Bus master applications available today demonstrate these benefits.

With the increasing sophistication in microcomputer applications in the future, the benefits of IBM's advanced bus masters will become even more significant and evident.

ABOUT THE AUTHOR

Dr. Patsy A. Bowlds is a senior technical consultant with IBM in Special Marketing Programs in Boca Raton, Florida. Her current position includes Micro Channel architecture marketing and technical support. Pat was formerly the manager of systems strategy development for the Boca Raton Laboratory. Her background includes engineering management positions in storage systems development and engineering/scientific technical support. Pat has also served on the Technology Staff for IBM's Entry Systems Division (ESD). Prior to joining ESD, Pat was with IBM's Information Products Division in printer supplies development in Lexington, Kentucky.

Overview of Extended Micro Channel Functions

*Chet Heath
IBM Corporation
Boca Raton, Florida*

In September of this year, IBM released information about extended functions of Micro Channel architecture. This article provides an overview of those functions. Further detail is provided in subsequent articles in this issue.

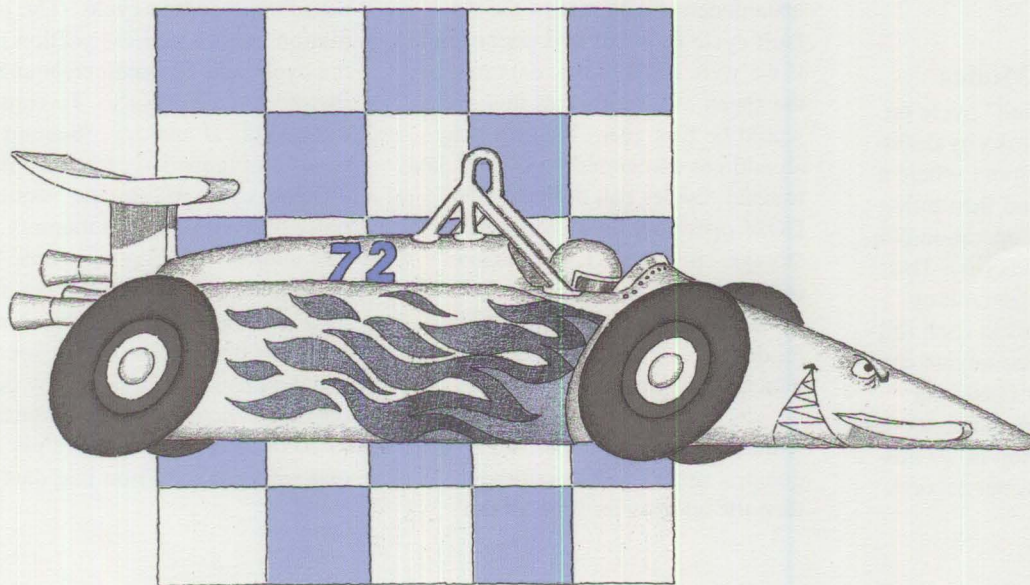
The functions that have been disclosed for the Micro Channel architecture standard advance its capabilities to communicate data and control operations between memory, I/O adapters, and processors in a wide variety of applications. Although these new functions will be introduced in systems as the implementations require, no architectural limit specifies high- or low-end exclusiv-

ity. The functions are optional extensions of the "basic" Micro Channel specification that are, in general, upward-compatible from existing design points – and in specific situations, they are also applicable to augment existing systems.

The Advanced Functions

- **Streaming Data Mode** – Improved efficiency in the use of time during data transfers yields a potential four- and perhaps eight-fold improvement in the speed of data transfer. While 80 MB data transfer is impressive, no adjustments to the physical structure of the systems, mechanical or electrical definition of the connector interface, or redefinition of existing protocols was required to achieve the increase. While not proven, the potential for 160 MB throughput is promising.
- **Parity checking** – Parity checking of data and/or address information during channel operations has been standard operating procedure in mainframes for many years. It is used to support the
- **Synchronous exception signaling** – This allows the association of the specific channel cycle with individual errors and enables error data collection to be synchronous with occurrence.
- **Subsystem Control Block (SCB) architecture** – This capability is similar to functions that have been available in IBM mainframe or minicomputer designs since 1965. It is the enabling software and hardware structure that can coordinate the operation of multiple bus masters in Micro Channel systems. It provides for system control of complex configurations with multiple intelligent subsystems and extends the foundation to support distributed multiprocessing in the future.

mainframe-like environment that is evolving for advanced microsystems with large memory systems, extremely fast I/O devices, and complex I/O configurations. Parity is typically employed in large-system architectures.



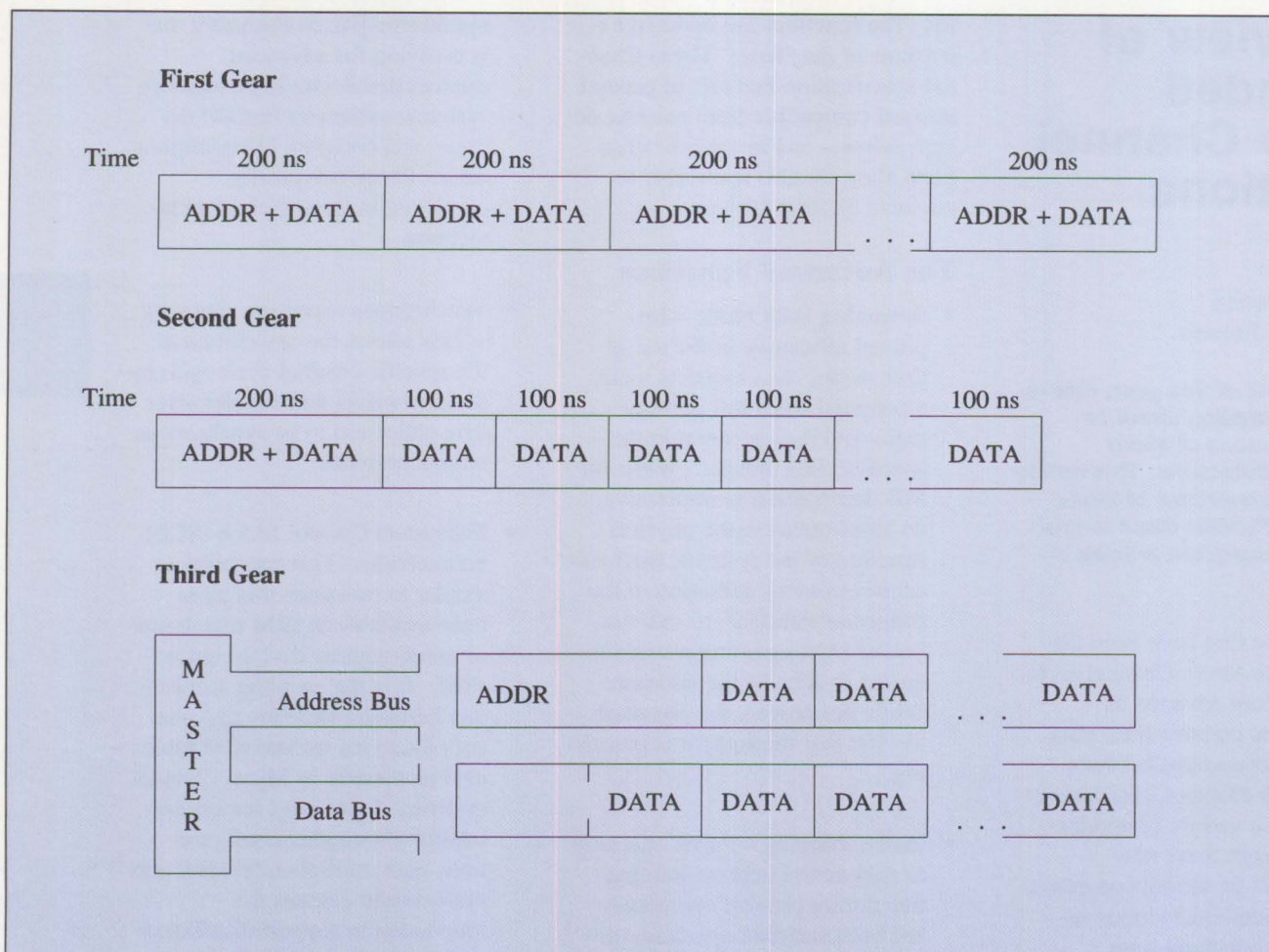


Figure 1. Streaming Data Mode

More specifically the functions are explained as follows:

Streaming Data Modes

The standard or "default" cycle on the Micro Channel works by defining the address in memory where a transfer is to occur, and then transferring the data. The operation consumes 200 nanoseconds (ns). This means that five such "default cycles" can be performed in each millionth of a second. Each cycle can move 4 characters, or bytes, per transfer; this yields an instantaneous rate of 4 times 5 million or 20 million characters (bytes) per second.

This is defining the "throughput," or instantaneous data rate of the default cycle to be 20 MB per second. If we were to liken this data rate to the speed of a sports car, this cycle would be first gear. By providing an address associated with each data transfer, cycles can define RANDOM operation by a processor on storage. In Figure 1, "First Gear" represents this graphically. A block of data is transferred, using burst mode as a "stream" of address and data transfer operations.

If the data is to be organized in sequential order in 32-bit storage, then the address portions of the

cycle will each be 4 bytes higher than the preceding cycle. The information in each address portion of the cycle will be predictable and, therefore, unnecessary. To continue our sports car analogy, "Second Gear" in Figure 1 represents a more efficient cycle to "stream" sequentially ordered data into memory. The address information would be presented once only in the beginning and thereafter maintained as a running count by the data's source and destination. This thereby doubles the efficiency of long transfers to 40 million-byte transfers per second with the condition that data typi-

cally be sequentially ordered in memory.

It may be observed that the 32-bit address bus is not imparting information after the first 200 nanosecond period. It is, therefore, available as an additional 4-byte path for data transfer. Four bytes of data moving on the address bus are coordinated with 4 bytes of data transferring on the 32-bit data bus. This yields 8 bytes of data transferred in one operation. This additional mode is called "Multiplexed Streaming Data Mode" and is capable of transferring sequentially ordered data at a rate of 80 million bytes per second. The mode is represented as "Third Gear" in Figure 1. This is important because rates that high can occur when many very high-speed devices (SCSI files, FDDI LAN) transfer data "concurrently" – all at the same time. It can also allow a block of data to be fetched from bus-attached memory fast enough to be usable for a high-speed memory cache elsewhere in the system.

Micro Channel architecture is enabled for additional procedures that will yield up to 160 MB. (Now you're in Fourth Gear!)

Random versus Sequential

Transfers: A faster cycle might have been designed around matched-memory protocol. When matched-memory protocol was originally defined in the technical reference

manual for PS/2 systems in April 1987, it used the asynchronous cycle capability of the Micro Channel interface to define faster cycles to bus-attached system memory.

Matched-memory protocol is simply a modification of the default cycle that can be accelerated to higher speeds, that is, with less time per period. With a 16 MHz, zero wait state system, the time for each period is reduced to 62.5 nanoseconds or one period of the 16 MHz processor clock (Figure 2). It yields a potential throughput of 32 million bytes per second. This is not nearly as fast as streaming data mode, but the data can be fetched randomly. Since no specific order is defined for the data, processor instructions can be fetched from high-speed memory on the bus at usable rates for the processor. There is then a trade-off between the ability to "stream" data at the highest speed and the ability to change the address randomly on every transfer. It is, therefore, important to classify the transfer type – random or sequential – whenever throughput comparisons are made. Random capability is very desirable when a processor fetches instructions and data from bus-attached memory; high-speed sequential data transfers are ideal for high-speed interface to fast I/O devices. A bus-attached processor, operating from a local cache, might use both modes to replenish the cache or control I/O. In

all cases, faster cycle times allow more time for other operations in the machine. The net is that the implementation will determine which is the best mode of transfer.

Parity Checking

The motto for the "MG" sports car was "safety fast." It wasn't sufficient to go fast from point "A" to point "B"; you also had to get to point "B" in one piece.

When we humans communicate, our oral data usually has meaning only in context. The words in a sentence often verify the meaning of other words. For example, if we say "the dog took several bytes out of my leg," the sentence makes no sense unless we exchange the word "bites" for "bytes." If we heard the sentence, we would assume the use of "bites," because it is the only choice that makes sense.

When computers exchange information, all combinations of data are typically valid; eight bits of data, called a byte, that define a data character, can yield 256 combinations of alpha, numeric, punctuation, and graphic characters. To put the data in context, a ninth bit can be added. The state – one or zero – of this ninth bit can be determined mathematically by counting the total number of ones in the data byte, regardless of position, and determining if it is odd or even (see *Note* at the end of this section). By calculat-

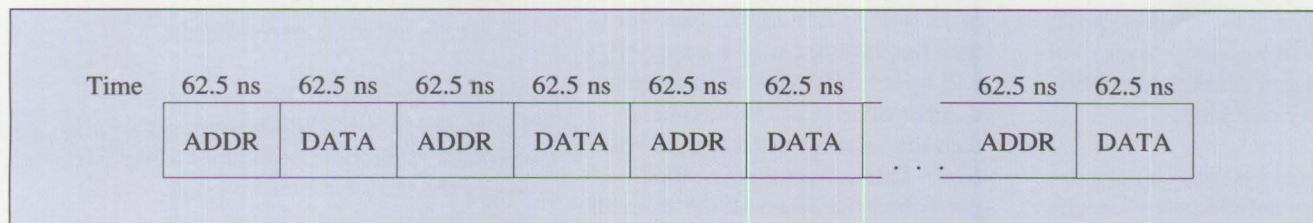


Figure 2. Matched-Memory Protocol


ing what the parity bit in the data should be and comparing it to what is actually received, the receiving party can determine if it has been sent good data.

Returning to the example of oral conversation, if someone with a heavy foreign accent were to say "the dog took several bites out of my leg," we might ask them to spell "bites" to verify the meaning; with others we might be more confident of what we have heard, and we would accept the verbal data without checking. We would be "adaptive" in that we would check the meaning for some, but not all, situations. What if the consequences of error were severe? One might imagine a large dog sitting next to a foreigner with a diskette in his mouth; is the foreigner warning us, or making a joke?

In the computer, some interfaces or adapters may have more severe reactions to distorted data than others. An error to the display might yield an invalid symbol in a word, or just one dot that is slightly off-color. Our brains would probably detect and either correct and ignore the error. However, data destined to be sent to a large data base in a mainframe file system might not be reused for months – or worse, it could be propagated into other files without ever being checked. If the data throughput is very high and the storage capability of the system is also very high, as in a mainframe, then the probability of an error increases significantly, and the need to check data on critical interfaces rises. For this reason large systems liberally employ parity checking.

Micro Channel systems can also be "adaptive" by defining error checking capability, only on a vulnerable interface. Alternately, parity could

be eliminated where limited consequences of error do not justify the expense. With adaptive parity, the bus master drives a special signal on the interface, which indicates that valid parity information is included in the data transfer. This signal allows the slave adapter to know if the transmitting party will include the ninth parity bit in the transfer or not. If it is there, then the receiving party can check the data. If an adapter receives parity, usually it also generates parity when it is the originating party in a data transfer.



The bus master drives a special signal on the interface, which indicates that valid parity information is included in the data transfer.

This definition of adaptive parity allows systems and options to be compatible, even if only part of the system, such as one master adapter and one slave adapter, implements the parity check logic. Furthermore, parity checking can be applied to data transfers and/or address information as well.

Note: By convention, if the number of 1s is an even number, like 4 or 0, then the 9th bit, called a *parity bit*, will be a 1. Conversely, if the total number of 1s is an odd number, then the value of the parity bit will be 0. This convention is called *odd parity* because when all the 1s in all 9 bits are counted, the number will always be odd. For example, if the

8 bits that make up the character byte of data are 0,0,0,1,0,1,0,1 or 1,1,1,0,1,1,0,0 each have an odd number of 1s, so the value of the 9th or parity bit is 0, yielding an odd number. By contrast, the 8-bit value 0,1,0,1,0,1,0,1 would yield a parity value of 1.

Synchronous Exception Signaling

When errors (for example, parity errors) occur in high-speed systems, delays in detection can cause status of the error condition to be presented after a transient condition that caused the error has passed. This is analogous to the situation of a burglar alarm in the bank. If the authorities arrive after the perpetrators have departed, the money may be gone forever. If, however, a guard stationed in the building can catch them in the act, the money can be retained, that is, the error and its source can be "logged," and the error causing condition can be, perhaps, arrested. In the computer it is the data that is valuable although it may represent the customer's money.

For example, if a communications interface is used to transfer data into a system, identifying an error in a long transmission would allow retransmission of just the segment containing the error and not the entire record. A similar example exists for tape input/output. Exact identification and reporting of the error condition may require detailed information on the state of the system at the instant the error occurs.

To carry the analogy further, providing a camera record of the events in the bank at the time of the loss can verify the facts for later investigation. In a very real sense, further similar events can be reduced if the

diagnosticians can pinpoint the events that lead to the error and prevent future similar "loss" situations. Synchronous exception signaling is then enabling the "arrest" of participants at the instant of potential loss, making collection of the evidence as to the nature of events leading to the potential loss of data or system control possible, and hopefully allowing full recovery of the data and the system from the situation.

This function is in addition to the asynchronous channel-check capability in the basic Micro Channel definition. The basic function works more like a "bank examiner" and will not necessarily identify the error but will alert the system that an error has occurred. It is saying, "the books don't balance at XYZ bank," allowing a log of errors and associated cards to be created.

Basic SCB Architecture Concepts: Control Blocks and Chaining

What it does: SCB architecture defines the hardware and software control means for:

- attachment of multiples of bus master attachments
- error and status reporting protocols
- dynamic assignment of system resources required to support concurrent activity of multiple masters.

It also introduces a powerful concept of "command chaining" where masters can decide, according to prescribed algorithms, which of several potential control paths to follow.

This allows the paths for coordination of bus master operations to be predefined as "macro" operations while giving the subsystems the latitude to respond to changing conditions at the time of operation. The SCB architecture greatly avoids the overhead of interrupt operation and anticipates the migration of mainframe operating system principles into the personal systems environment.

*"Command chaining"
allows masters to
decide which
of several potential
control paths
to follow.*

Control block architectures have been defined for IBM mainframe and minicomputer products for approximately 25 years. However, personal computers, including the PC, PC XT, and AT series of products, depended on interrupt requests and the Basic Input/Output System (BIOS) as the interface between the operating system and I/O adapters. Interrupts and BIOS allowed adapters to evolve to more powerful designs but still placed a heavy burden on the processor.

Avoids Interrupts: Systems with processors designed for rapid interrupt handling are called real-time systems and typically employ a separate set of processor registers for each interrupt level. This is one area where the single-tasking legacy of the personal computer is most telling. Typically, it can take 200

processor instructions in these single-tasking machines to respond to an interrupt. One thousand or more instructions per interrupt are not uncommon for a multitasking operating system. Why so many? Without a real-time interrupt structure, it can take at least that number of instructions to save the environment (environments, in a multitasking operating system), to operate the I/O and restore the original environment.

If only one user and one task are active at a time, as in most PC applications under DOS, processing will wait on slow I/O. In this environment, large interrupt overheads can be hidden by the I/O delays. The existing scheme for operating system to I/O interface is inadequate as we move to a "multi" environment, where I/O adapters can be as concurrently active as the tasks that stimulate them. The cumulative overhead from many concurrent interrupts can severely load a system's performance. For example, if you want to increase the performance of an asynchronous communications interface, what do you do?

Most often the solution has been to keep the asynchronous adapter card and the communications application, and replace the system processor. This increases the rate at which the communications application runs. If a 6 MHz machine won't do, then get an 8 or 10 or 12 MHz system to run the asynchronous adapter faster. Soon we have a lot of money tied up in expensive memory to hold the application for fast processors to run the serial adapter at a faster rate. A better solution to this problem is to employ a bus master to control the serial interface and maintain the user's investment in the system unit. This

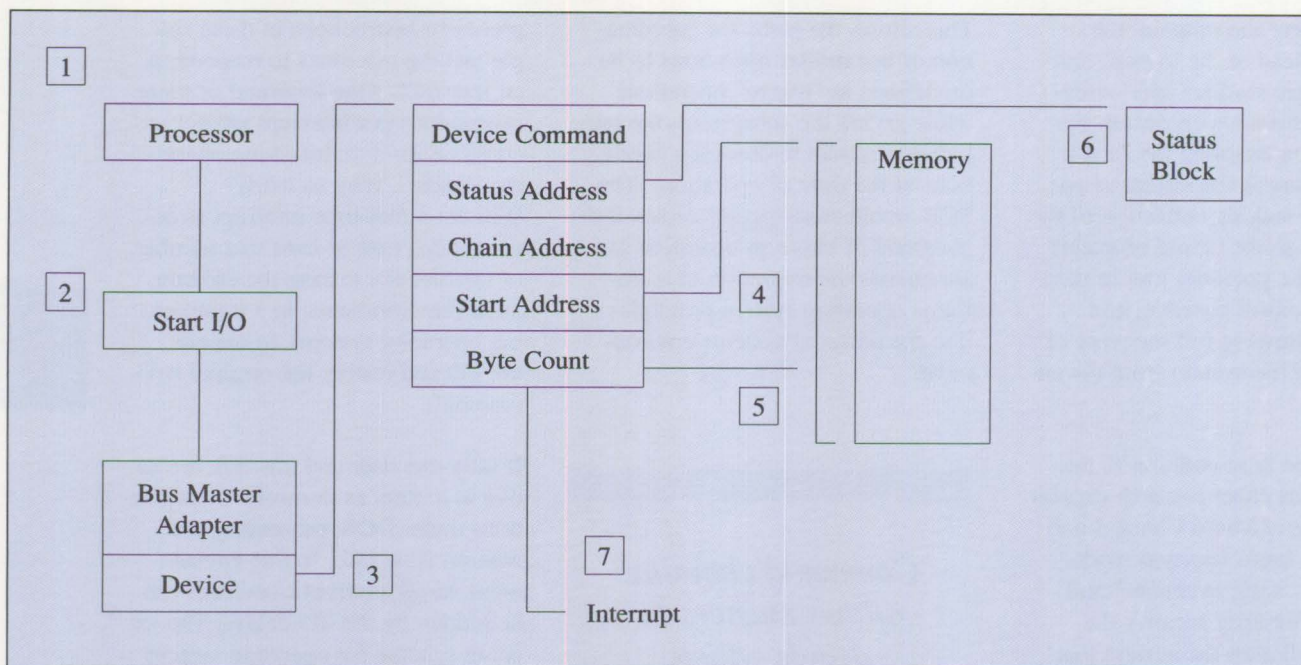


Figure 3. Process of a Generic SCB

releases the processor for operating system and applications.

But what about the software interface to control bus masters?

So far, discussion about the feasibility of bus masters in systems has focused on arguments of throughput to main storage, efficiency of arbitration, or what the best algorithm is to ensure that each adapter gets a fair share of the channel time.

Many of these arguments center on assumptions about system structure and elevate the importance of performance over issues of system integrity, error recovery, diagnostics and intersystem interoperability. These are the real issues! The software interface between the operating system and I/O adapters is the key to the resolution of those issues.

Builds in the future: Subsystem Control Block (SCB) architecture can enable the design of ultra high-speed, peer-to-peer communications

between adapters and simpler and faster operating systems, and can reduce much of the burdensome interrupt overhead that is the real bottleneck to performance. SCB architecture is defined to allow intelligent bus master systems to work together smoothly and to allow a graceful evolution to mainframe-like operating systems in workstations and personal systems. And that IS the future!

Quite simply, the SCB architecture depends on the ability of the Micro Channel bus master adapter to randomly access and control memory or I/O directly, rather than requiring the processor to move data or control the adapter.

Frees the processor: With SCBs, the processor creates a block of data in common memory where a bus master can read it and follow the commands defined in the block. A generic SCB might work as shown in Figure 3.

1) The processor prepares an SCB (it may be prefabricated and stored on a disk or be constructed on the fly).

2) The processor issues commands to the bus master adapter, indicating the location of the SCB in memory.

3) The bus master arbitrates for control of the system and fetches the SCB. The command from the SCB can be delivered by the adapter to the device without processor overhead.

4) The bus master then transfers data with memory starting at the address specified in the Start Address field.

5) The transfers continue until the length of the data specified by the byte count has been reached.

6) Should an error occur, recovery is facilitated by automatically writing the adapter and/or device status

to the address specified in the Status Address field.

7) After all operations are complete, the processor is interrupted to indicate the adapter is now free for another command.

If this is all that SCB architecture could do, it would be an improve-

ment over interrupts and BIOS because it would typically amortize the interrupt overhead over many data transfers and free the processor to perform other duties after the initial command. Yet SCB architecture can do more.

A much more powerful function called "command chaining" is en-

abled by SCB. The operation begins as in the simple case above, but omits step 7 and continues on to steps 8 through N (Figure 4).

What SCB Chaining Can Do:

SCB data chaining can allow the concatenation of data buffers that have been fragmented by virtual addressing, and can define macro operations, like background backup, and

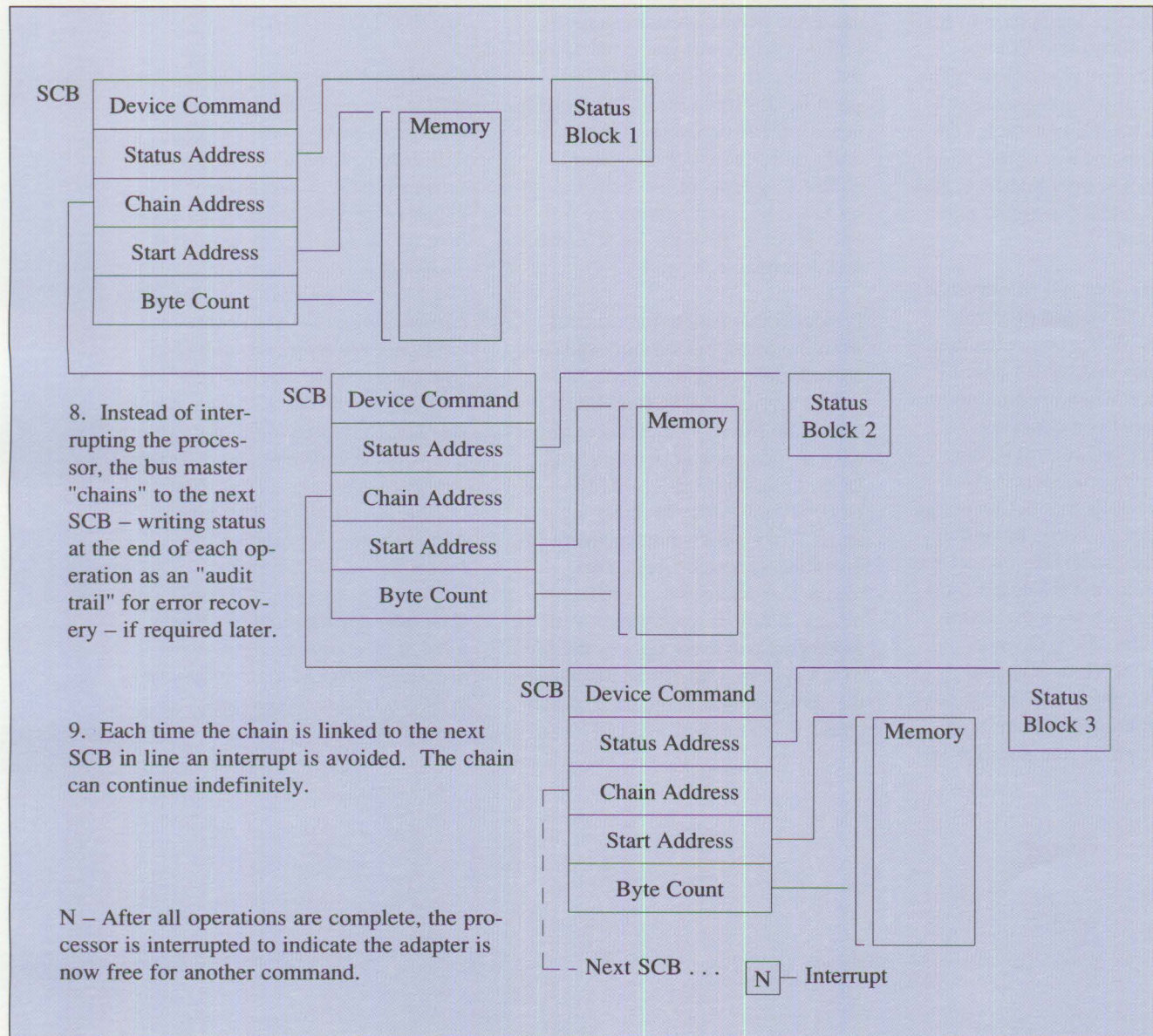


Figure 4. Command Chaining

device-to-device offline copy. It can enable many powerful functions that do not involve the processor except to initiate and terminate operations or to handle recovery procedures should an error occur. The performance of the system as a whole is not dictated by the capability of the processor that originally shipped with the system unit; this protects the user's original investment and allows the user more freedom to customize the system to his needs! The Subsystem Control Block architecture is enabled by the Micro Channel bus master capability, and it is itself an enabling function for distributed multiprocessing on the Micro Channel interface, and that portends great functional extension and power.

Significance: We call Micro Channel a "channel" instead of a "bus" for a reason. A bus is a collection of data signals that have a coordinated function when operated in unison. A channel is a collection of busses, associated control signals used in transfer procedures AND software protocols that define operation in a system. While the initial Micro Channel specification met the latter definition, the disclosed capabilities greatly advance the definition of how the Micro Channel interface will work in a system. While the extension further defines the "rules" that govern data movement in the system, the designer is

offered more, not less, latitude in design alternatives.

It is important to remember that the Micro Channel interface has already supported designs that attach 8088, 8086, 80188, 80186, 80286, 80386 and 80486 Intel processors. Attachment of Intel i860 RISC and IBM RT® RISC processors, Motorola 68030, and IBM 370 processors have been implemented as well. Because the Micro Channel supports both asynchronous and synchronous data transfer, and because it has great flexibility in throughput, error detection and recovery, it is essentially independent of processor choice. Systems of like and dissimilar processor combination are not only feasible; they can be affordable and dependable as well.

Protocols that isolate the channel from the operating system can now be well-defined and extended as required through the SCB architecture. The Micro Channel computers have demonstrated operation with DOS, OS/2, OS/2 EE, VM386, UNIX®, and 370 VM operating systems. They are also independent of operating system choice. This means that the user has a maximized latitude to configure a system to varying needs throughout the life of the system, rather than having the choices fixed at the time of purchase.

Enhanced design latitude, extendability, and adaptability allow the user to maintain the value and capability of the system over time. This is, of course, good news. The best news of all is that the advanced functions have been disclosed for Micro Channel architecture without altering or obsoleting the originally published specification.

ABOUT THE AUTHOR

Chet Heath is a senior engineer at IBM's Entry Systems Division laboratory in his twentieth year with IBM. He holds a bachelor of science degree in electrical engineering from the New Jersey Institute of Technology and a master of science degree in electrical engineering from the IBM LSI Institute at the University of Vermont. He refers to himself as "the oldest living survivor of Micro Channel architecture" because he was involved in its definition since inception and gave the architecture its name. He has received IBM Quality, Outstanding Technical Achievement, Outstanding Innovation, and Corporate Technical Achievement Awards. Chet also has attained the eighth level of invention awards. He is presently assigned to development of Micro Channel strategic enhancements.

New Micro Channel Features

James Nicholson and
Fred Strietelmeier
IBM Corporation
Austin, Texas

Some new features of Micro Channel architecture that increase the data transfer rate and enhance the data integrity of the bus transfers were recently disclosed. These new features include:

- Streaming data procedure (including 64-bit transfers)
- Address and data parity
- Synchronous exception signaling.

This article provides a technical overview of these features of Micro Channel architecture.

Streaming Data (Including 64-Bit Transfers)

Streaming Data is a procedure that provides the ability to transfer multiple data cycles within one bus envelope. The procedure distributes the device selection overhead across the total packet, nearly doubling (for 32 bits) or quadrupling (for 64 bits) the performance capability of the Micro Channel bus. Sustained performance is a function of the total packet length and is illustrated in Figure 1.

Figure 2 illustrates the procedure used in basic transfer and 16- and 32-bit streaming data cycles. A basic transfer cycle is initiated by a bus master placing an address on the bus, and then activating one of the status S(0,1) signals. A typical bus master allows 100 nanoseconds (ns) for device selection, then in an-

other 100 ns period it activates the command (CMD) signal to control the data transfer. A streaming data cycle is similar, but a block of sequentially ordered data is transferred during the command period. The address specifies only the starting address of the block.

Three new signals, Streaming Data Request 0,1 (SDR(0,1)) and Streaming Data Strobe (SD_STB), mediate 16- and 32-bit streaming data cycles. The SDR(0,1) signals are driven by the bus slave following activation of the Address Data Latch (ADL) signal to indicate that it supports streaming data. The SD_STB signal is driven by the bus master and serves as a clock delimiting words in the data transfer. Each negative transition of the SD_STB signal defines a new data transfer of 2 or 4 bytes. At the end of a streaming data cycle, the deactivation of the CMD signal serves as the last bus clock.

One additional signal, Multiplexed Streaming Data Request (MSDR), is

required for 64-bit streaming data. Figure 3 illustrates the 64-bit streaming data cycles. The MSDR signal is driven by the bus slave concurrently with the SDR(0,1) signals to indicate that it supports 64-bit streaming data. The bus master indicates its support of 64-bit streaming by tri-stating the address bus and driving the Byte Enable (BE(0-3)) signals inactive (high) after the activation of CMD and the detection of an active MSDR signal. The address bus is then used with the data bus to transfer 64 bits of data with each SD_STB as described above for 16- and 32-bit streaming data.

The SDR(0,1) signals also provide for speed matching between the bus master and slave. The SDR(0,1) signals are coded to provide information about performance characteristics of the slave. This information is used by the bus master to ensure that it does not exceed the maximum clocking rates of the slave. Figure 4 provides the cur-

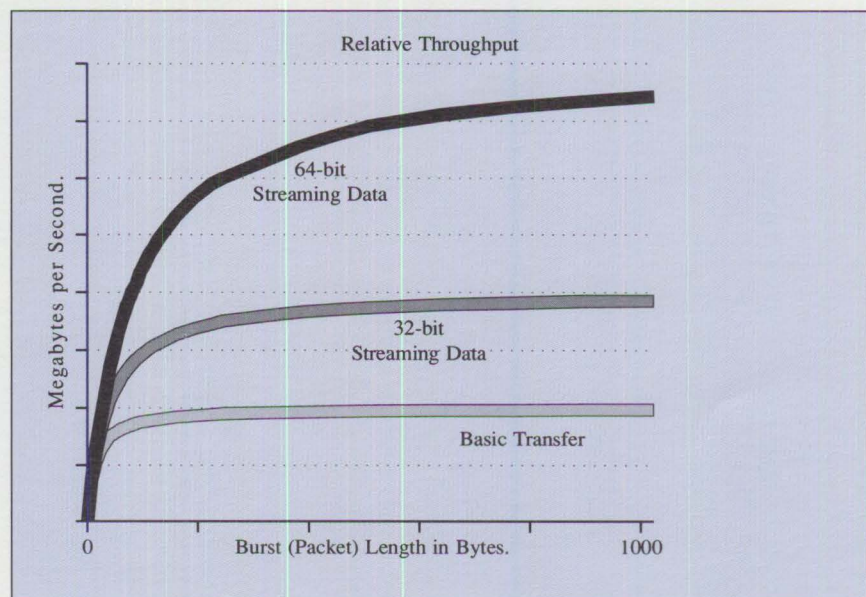


Figure 1. Streaming Data Performance Improvements

rently defined encoding for SDR(0,1).

The SDR(0,1) and MSDR signals do not have to be valid for a bus

master to initiate a streaming data cycle. Instead, a bus master normally activates the SD_STB signal concurrently with the CMD signal and samples the SDR(0,1) and

MSDR signals just before the deactivation of the SD_STB signal. If the SDR(0,1) and MSDR signals are not active at that time, the bus master deactivates the S(0,1) signals to

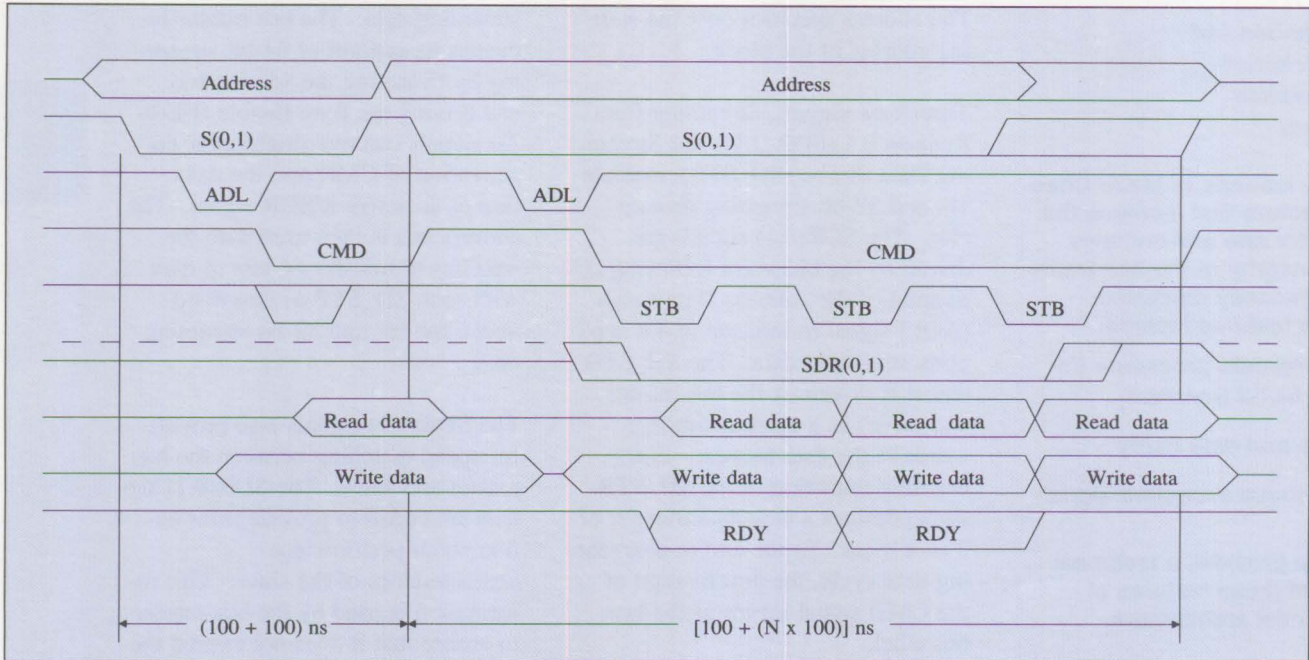


Figure 2. Streaming Data – 16- and 32-bit Transfers

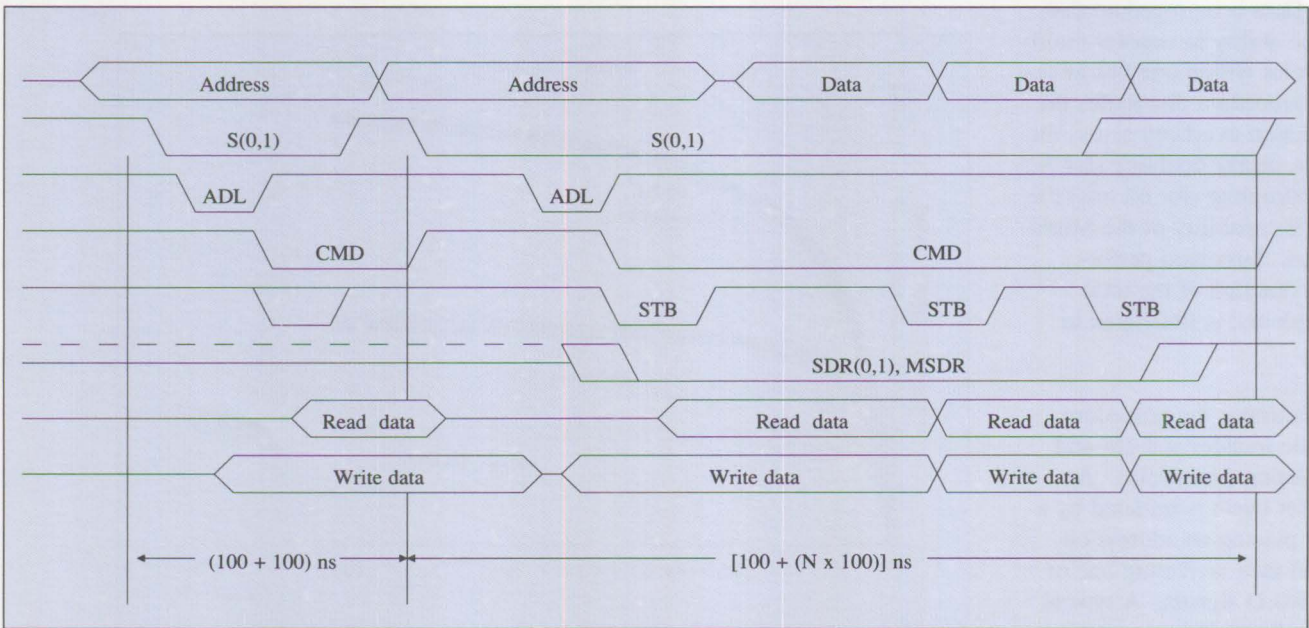


Figure 3. Streaming Data – 64-Bit Transfers

- SDR		Resulting Cycle
0	1	
0	0	Reserved
1	0	Reserved
0	1	10 MHz Max (100 ns min)
1	1	Basic Transfer Cycle (200 ns min)

Figure 4. Currently Defined Code Points for SDR(0,1)

complete the cycle as a basic transfer. This provides relaxed timing requirements for the SDR(0,1) and MSDR signals. The SDR(0,1), MSDR and SD_STB signals use tri-state drivers and require resistive pullups on card inputs to hold the signals high when not being driven.

The Streaming Data procedure for 16- and 32-bit transfers includes synchronous data pacing using the Card Channel Ready (CD_CHRDY) and Channel Ready Return (CHRDYRTN) signals. (This capability is not available on 64-bit streaming data transfers.) The slave indicates that it cannot complete a particular data cycle by deactivating the CD_CHRDY signal after the SD_STB signal is deactivated. This is propagated to the CHRDYRTN signal by a logical AND on the system board for sampling by the bus master. The bus master is required to repeat the cycle if the CHRDYRTN signal is inactive (low) on the next transition of the SD_STB. The SD_STB signal must continue cycling until the slave activates the CD_CHRDY signal.

Termination controls are symmetrical so that either the bus master or slave can end a streaming data cycle. The procedure is synchronous and is illustrated in Figure 5. Bus masters request termination by synchronously deactivating the S(0,1) signals coincident with activation of the SD_STB signal, whereas slaves request termination by synchronously deactivating the SDR(0,1) and MSDR signals following the activation of the SD_STB signal. Bus masters should not terminate the cycle until the SDR(0,1) and MSDR signals are deactivated. If a slave is not ready when a master deactivates the S(0,1) signals, the slave should set the CD_CHRDY signal low and hold the SDR(0,1) signals active until the cycle in which the CD_CHRDY signal will be set high. Although not illustrated, the architecture supports the ability to defer initiation of streaming data cycles.

For compatibility reasons, all 16- and 32-bit streaming data cycles must be initiated on 4-byte address boundaries. The reason is that Micro Channel architecture includes a byte-steering function to facilitate transfers between 16-bit bus masters and 32-bit bus slaves. This steering

function is latched at the time the CMD signal is activated and remains static throughout the data transfer envelope. As such, it would not properly steer the data when operating with 16-bit streaming masters. Specifying streaming data to begin on 4-byte address boundaries disables the steering controls, but results in the requirement that 32-bit slaves provide their own byte steering when engaged in data streaming operations with 16-bit masters. Streaming data cycles of 64 bits must begin on 8-byte address boundaries.

Address and Data Parity

Address and data parity are provided to improve the data integrity characteristics of Micro Channel architecture. These mechanisms are particularly well suited to detecting typical errors such as card seating problems, power disturbance, and electromagnetic interference. Address parity and data parity support are optional and independent, and mixtures of parity and non-parity devices are permitted. The parity supported in Micro Channel architecture is odd parity. Figure 6 illustrates the bus signal procedures for parity.

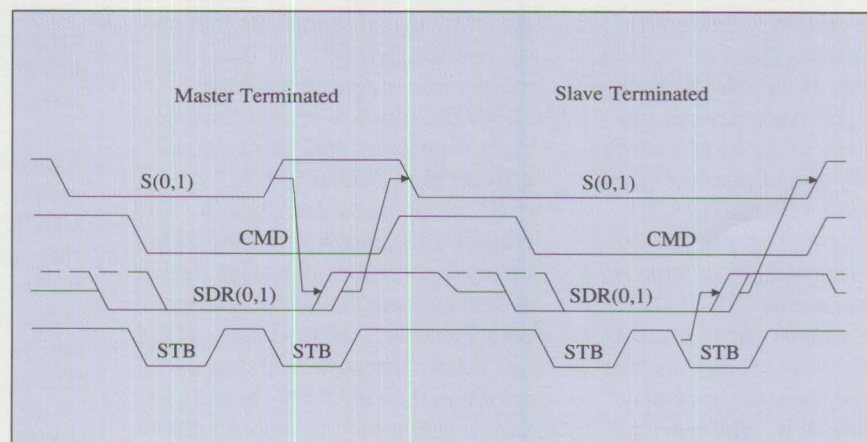


Figure 5. Streaming Data

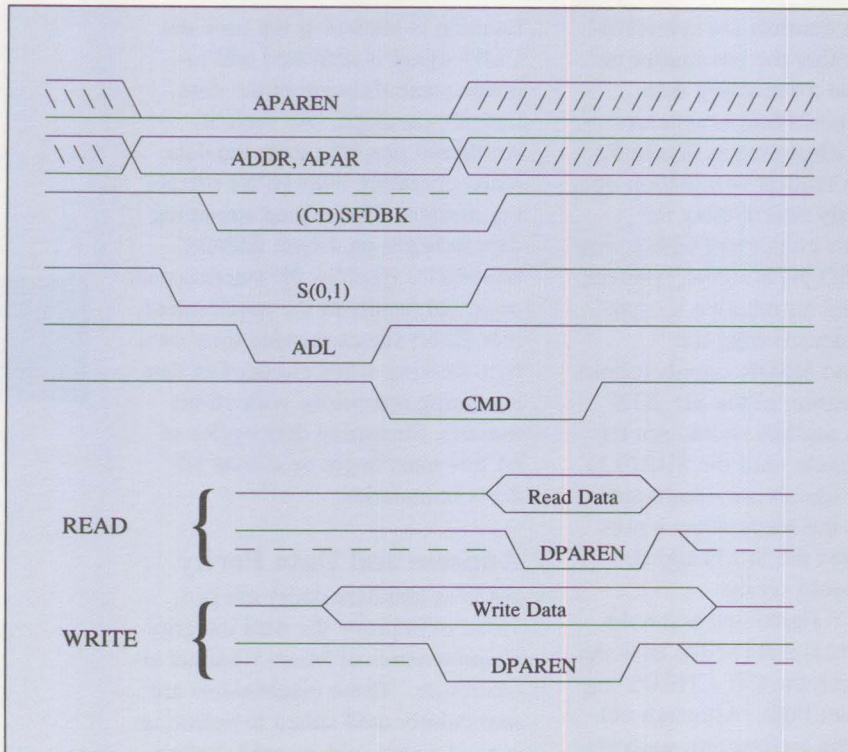


Figure 6. Address and Data Parity

Address parity is controlled by the Address Parity Enable (APAREN) signal driven by the bus master. Normally, a slave device responds to a valid address by activating the Card Selected Feedback (CD_SFDBK) signal. This is propagated to the Select Feedback Return (SFDBKRTN) signal by a logical AND on the system board, notifying the bus master that a device has been selected. If the APAREN signal is active, all slave devices supporting address parity should check address parity and, if incorrect, activate the Channel Check (CHCK) signal as described later in this article. A slave should not become selected or activate the CD_SFDBK signal if the address parity is incorrect and should not set any internal error status because the error may not have affected it. The bus master is then expected to suspend operations, set internal error status

indicating that a channel check occurred with no SFDBKRTN, and interrupt the system.

Data parity checking is controlled by the Data Parity Enable (DPAREN) signal. During write operations, the bus master activates the DPAREN signal coincident with the CMD signal to indicate that parity is associated with the data. If a parity error is detected and the DPAREN signal is active, slave devices supporting data parity should activate the Channel Check (CHCK) signal as described in "Synchronous Exception Signaling." During read operations, the slave device should activate the DPAREN signal coincident with the read data. If the bus master detects a read data parity error and the DPAREN signal is active, it is expected to suspend operations, set internal error status indicating that a read parity error oc-

curred, and interrupt the system. The CHCK signal is not normally activated on a read-data parity error. Following the deactivation of the CMD signal, the slave device is required to provide an active restore of the DPAREN signal by driving the signal high before tri-stating it.

Slave devices supporting parity in internal arrays can propagate the internal parity bits onto the I/O bus to allow synchronous detection of the error by the bus master. The slave should also check parity on the array data and set any errors into an internal status register. The system error handler can then use this status register to differentiate between array and bus errors. Refer to the next section for details.

Synchronous Exception Signaling

Micro Channel architecture provides for exception signaling with the CHCK signal. Initial use of the CHCK signal in the Personal System/2 line of products was asynchronous. Use of the CHCK signal may support the signaling of synchronous events in some processors, such as write parity errors, page faults, and command queue overflows. This enhancement is key to meeting system data integrity and error recovery requirements, and has utility in support of high-performance graphics display adapters.

Synchronous exception presentation simply provides for activation of the CHCK signal in the same bus cycle that caused the exception. Normally, the CHCK signal will be activated while the CMD signal is active, as illustrated in Figure 7. The architecture specifies the timing of the CHCK signal relative to the CMD signal to ensure proper operation.

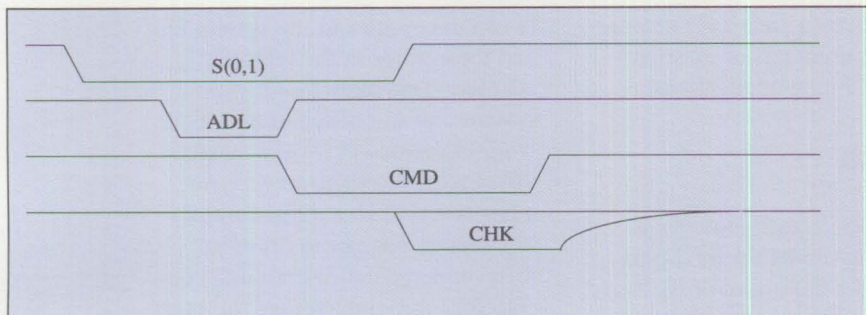


Figure 7. Synchronous Exception Signaling

Presentation of write-data parity errors during streaming data operations requires slightly different timing. The reason is that the slave receives the last data packet when the CMD signal is deactivated, and additional time is required to validate parity. To accommodate this, the architecture provides for a late presentation of the CHCK signal with the DPAREN signal. Bus masters are required to hold the DPAREN active for a short period of time following a streaming-data write operation, and this signal should be used by the slave to enable activation of the CHCK signal.

Conclusion

Three features of Micro Channel architecture have been discussed. These features broaden the characteristics of that architecture to support advanced I/O devices or systems. Improvements include:

up to a four-fold increase of bandwidth capabilities, addition of parity checking as an optional feature, and support of synchronous exception signaling. The enhancements use previously reserved pins on the Micro Channel bus and retain compatibility with existing card and system board designs.

ABOUT THE AUTHORS

James O. Nicholson is a senior engineer in the hardware architecture area. He received a bachelor of science degree in electrical engineering from the University of Minnesota and joined IBM in Rochester, MN in 1967, where he assisted in the development of eight systems I/O products. He transferred to Austin, Texas in 1979 and worked on the 5260 point-of-sale terminal and the

IBM RT family of products. He has also been closely involved in the development of Micro Channel architecture and received a corporate award for that activity in 1988. Jim has ten patents filed and fifteen publications in the "IBM Technical Disclosure Bulletin." He has received four formal IBM awards, three invention achievement awards and a number of informal awards. The formal awards include two Outstanding Achievement Awards, an Outstanding Innovation Award, and an Outstanding Contribution Award.

Fred E. Strietelmeier is a senior engineer/manager in the hardware I/O architecture area of IBM's Advanced Workstation Division. He received a bachelor of science degree in electrical engineering from Rose Hulman Institute of Technology and a master of science degree in electrical engineering from Rice University. He joined IBM in Austin, Texas in 1972 with initial responsibilities in circuit design. He has worked on dictation products, IBM DisplayWriter family of products, and most recently, the RT family of products. Fred had management responsibility for the development of Micro Channel architecture. He received an Outstanding Technical Achievement Award for this activity.

SCB – An Architecture for Micro Channel Bus Masters

*Frank Bonevento, Ernie Mandese, Joe McGovern and Gene Thomas
IBM Corporation
Boca Raton, Florida*

The Subsystem Control Block (SCB) architecture was developed by IBM to standardize the engineering task of designing Micro Channel bus master programming interfaces as well as the programming task of supporting them. This article provides an introduction to the architecture, discusses some of its underlying concepts, and describes its delivery service facilities.

The Micro Channel bus master facility provides a structure for the independent execution of work in a system. Independent execution of work by bus masters frees a system unit, for example, to perform other tasks. Doing more work in parallel typically improves the response time and throughput of the system as perceived by the end user.

The SCB architecture builds on the Micro Channel bus master capabilities by defining the services and conventions needed to design, implement, and use bus masters effectively. The SCB architecture supports many functions found in larger IBM systems designed to facilitate multiprocessing. Command chaining, data chaining, signaling masters, and a free-running duplex control block delivery service have been provided. Finally, the ability to support user-defined control

blocks provides a means for dealing with the requirements of existing, current, and future applications.

Purpose

The SCB architecture provides a programming model for the Micro Channel and a definition of the logical protocols used to transfer commands, data, and status information between bus master feature adapters. The SCB architecture employs Micro Channel architecture as its underlying physical transport mechanism.

The term "control block" in the architecture name refers to the organization of the control information (commands) into areas that are separate from the data areas. The separation of control and data is used to increase performance, raise the level of functional capability, and provide the implementation independence required by today's applications.

The SCB architecture defines a control block structure for use between entities in a system unit and entities in feature adapters that have Micro Channel bus master capabilities. "Entity" in this context is a collective term referring to both device drivers and/or resource support found in a system unit, as well as device interfaces and/or resources found in feature adapters. The architecture also defines how control block delivery is provided between entities in a system unit and a feature adapter, as well as between entities in feature adapters on the Micro Channel.

The contents of the control blocks have been defined so that they offer a great deal of functional capability while, at the same time, providing implementation independence be-

tween the entities in the system unit and the entities in the feature adapter. This allows entities in a system unit to offload or distribute work to entities in feature adapters, thereby freeing them to perform other tasks. It also allows entities in a feature adapter to optimize their implementations without concern for entity interactions or internal implementation details.

The level of interaction between entities of the architecture is at a logical protocol level in order to insulate users from the details of, and interactions with, the underlying implementations. This means that applications written to adhere to the architecture and the underlying implementations that conform to it will remain viable even as technology advances.

Thus, as a feature adapter or system unit is enhanced to take advantage of new technology (higher data rates on the Micro Channel, the existence of data and/or address parity checking, and so on), it will be possible to continue to use existing applications with little or no change.

Scope

The SCB architecture has been designed to have broad application and be used in a variety of areas, including the following:

Traditional I/O Protocols: Traditional I/O protocols typically have a single system unit that requests a feature adapter to perform work on its behalf. This type of feature adapter might be found in an intelligent file subsystem on a personal system or a workstation.

For the traditional I/O systems, application of the architecture would

probably mean use of the Locate mode form of control block delivery. In Locate mode, only the address of the control block is delivered to the subsystem. The subsystem uses this address to locate the control block and to fetch it into its own private storage area for execution. In these I/O systems, a close synchronization is usually maintained between the request and the response to the request.

Peer-to-Peer Protocols: In peer-to-peer protocols, requests flow not only from the system unit to feature adapters, but also from feature adapter to feature adapter without direct involvement of the system unit. This might be found in feature adapters that serve as local area network (LAN) bridges or gateways, providing routing for file servers on LANs. Peer-to-peer protocols may also be found in loosely coupled multiprocessing systems.

Communications Protocols: Communications protocols may require routing to network servers within a feature adapter and the handling of replies that arrive out of sequence and are interspersed with requests. They may also require the ability to handle the movement of large amounts of data at very high speeds.

Response-Time-Critical

Applications: Response-time-critical applications require fast response time. The architecture defines a full-duplex, free-running delivery capability to support multiple work requests and replies. The amount of data transferred as well as the speed of the data transfer are critical in this environment.

In the last three categories, application of the architecture would probably mean the use of the Move mode form of control block delivery. In

Move mode, control blocks are moved from the requesting, or client, entity to the providing, or server, entity using a shared storage area. This shared storage area behaves like the named pipe function found in several of today's operating systems. Delivery using pipes is free-flowing and defined so that separate pipes exist for both inbound and outbound flows to and from the server (full duplex operation). This form of delivery is commonly used by controllers that have very high arrival rates of work requests and/or run in an asynchronous manner.

The SCB architecture is intended for use by both IBM and non-IBM developers designing bus master feature adapters for the Micro Channel.

Architecture Overview

The following sets forth some basic concepts used in developing the SCB architecture, its hardware context, system context, and its service structure.

Concepts

In the personal systems and workstation environments, there is a need to establish a higher-level, control block-oriented interface between support operating in a system unit (system-side entities) and support operating in a feature adapter (adapter-side entities). There is also a need for the same higher-level interface between support operating in two different feature adapters.

The nature of this interface is such that the control block information and the data can be considered as two separate parts of the communi-

cation between two cooperating entities.

Control block delivery service is used by each entity to communicate control information. Based upon this control information, there may also be data to be communicated between the two entities. This is referred to as "data delivery." Additionally, there may be information required during system initialization to tailor the delivery support to a particular configuration. This is configuration information.

Figure 1 gives an overview of the control block, data, and configuration delivery support.

Capabilities

Understanding the SCB architecture starts with understanding the requirements that are common to designers of Micro Channel bus master hardware and engineering software. From these requirements, the capabilities of a bus master feature adapter can be identified. The following are some of the most commonly requested capabilities.

Provide a Programming Model for the Micro Channel:

- Provide flexibility
- Support function distribution
- Support signaling among all bus masters and their users
- Provide a higher functional level of interface
 - Support command chaining
 - Support data chaining

- Provide request / reply protocol between users
 - Provide means of correlating requests to replies
 - Provide source and destination identification
 - Allow multiple outstanding requests
 - Allow unsolicited operations
 - Allow data to be carried within control blocks
- Support Bus Masters on the Micro Channel:**
- Provide a processor-independent architecture
 - Provide full duplex operation
 - Support feature adapter-to-feature adapter operation (peer-to-peer)
 - Support asynchronous operations

Improve Performance and Throughput:

- Reduce interrupt overhead
- Support expedited requests

Hardware Context

To better understand the structure of control blocks and the overall structure of the control block delivery service, it may be helpful to provide an example of the hardware within which the delivery service is expected to operate.

Figure 2 is an example of a hardware configuration that would benefit from using control blocks and the control block delivery service. This example shows several types of bus master feature adapters present on the Micro Channel, each using the bus master capabilities of the Micro Channel, and each providing support for one or more resources or devices. The example shows that there are multiple types of I/O system support within the system unit, each providing access to and support for resources and/or devices associated with a particular subsystem and feature adapter.

The functions available on bus master feature adapters are also used to establish peer-to-peer relationships between feature adapters, as well as between system units and feature adapters.

System Context

It is helpful to have an example that shows where control blocks and control block delivery fit relative to a system unit or feature adapter operating environment. This can easily be done for a system unit, but is much more difficult to do for the many different types of feature adapters. This is due to the nature

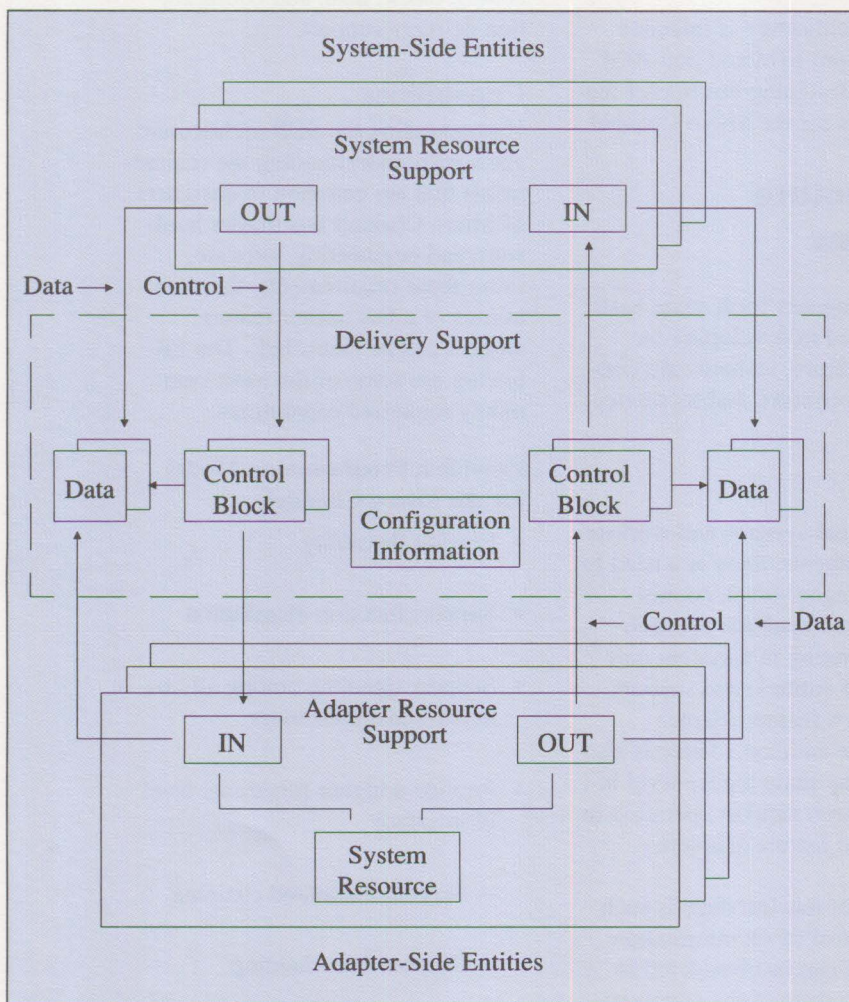


Figure 1. Overview of Delivery Support

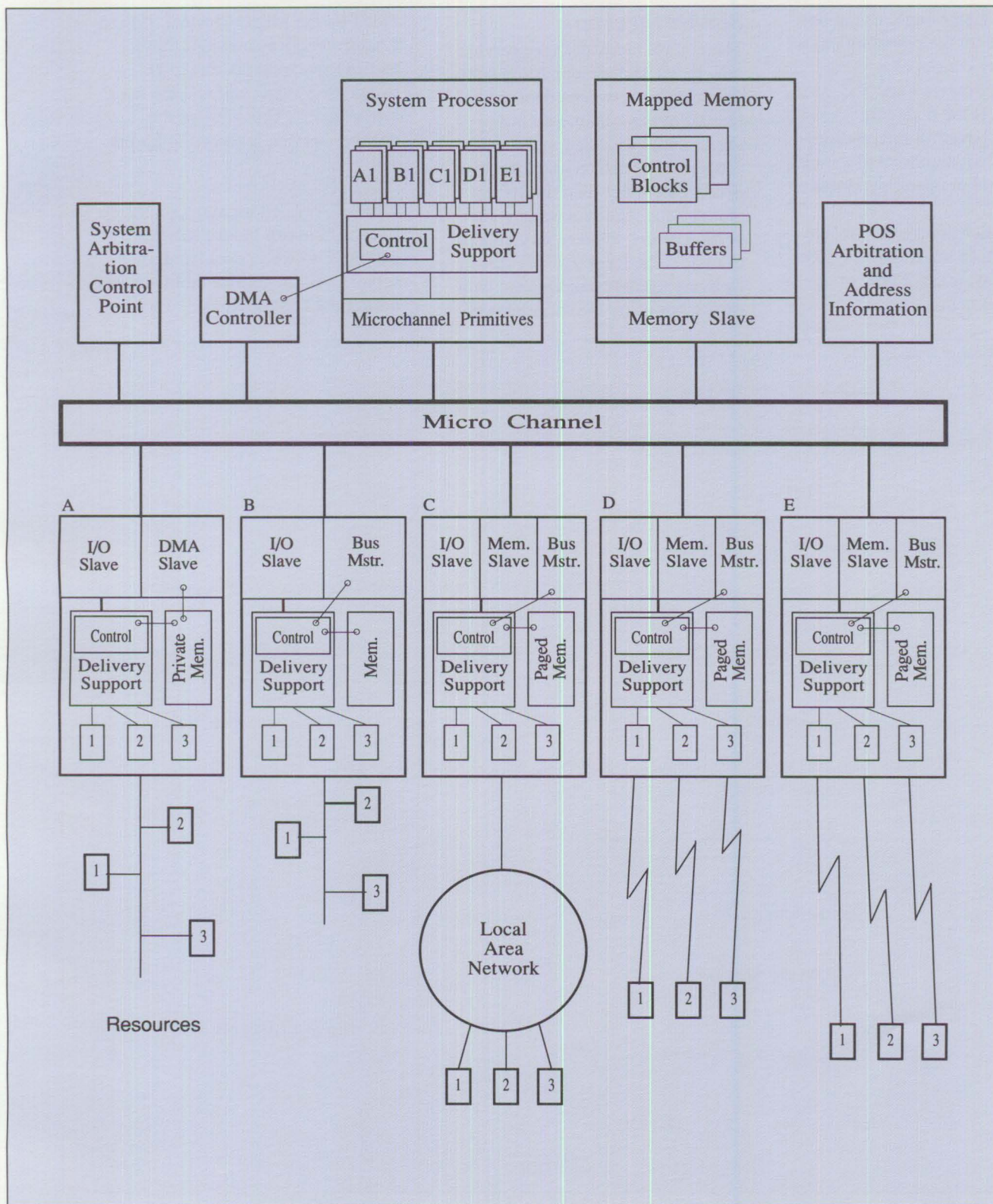


Figure 2. Hardware Environment

of feature adapter implementations. At the low end, the operating environment may consist of nothing more than hardware logic and state machines. At the high end, it may consist of a powerful microprocessor with paged memory and a multi-tasking kernel or operating system.

Figure 3 is an example of how the delivery service maps into a system unit in a DOS or OS/2 type of operating environment.

Service Structure

The control block delivery service may be viewed as supporting communications between client entities located in the system unit and server entities located in a feature adapter. The delivery service itself is distributed between the system unit and the feature adapter. The portion of the delivery service local to each is the local delivery agent. Delivery agents communicate with each other using the services pro-

vided by the Micro Channel. Micro Channel services include memory and I/O space that is shared between the system unit and the feature adapter. This view of the delivery service is shown in Figure 4.

The delivery service supports the delivery of control blocks between pairs of entities (a client and a server). These entities build and interpret the control blocks. The ac-

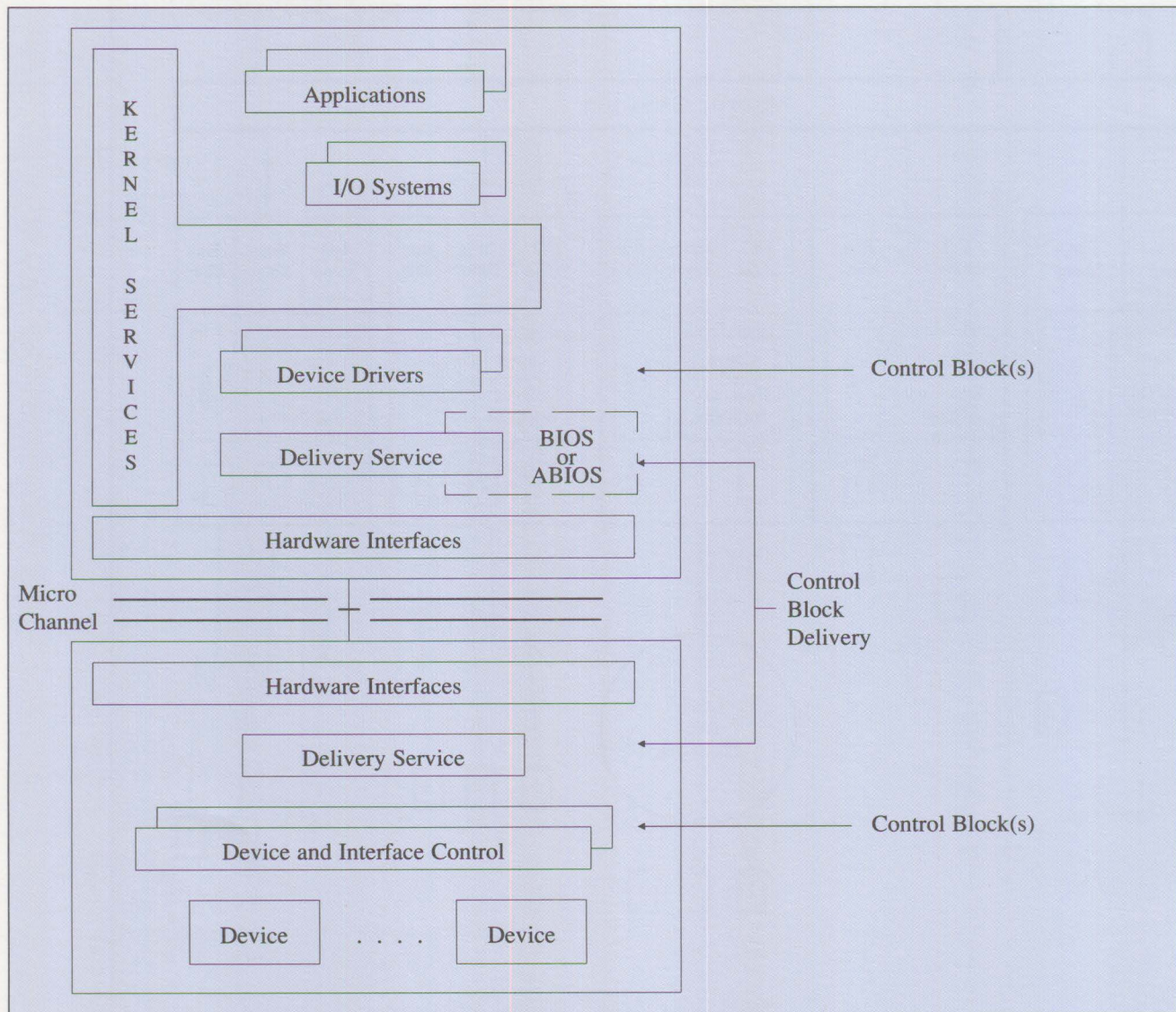


Figure 3. Example of System Unit Operating Environment

tual control blocks, their content, and their sequence determines the specific entity-to-entity protocol being used between a client and server.

Both the control block delivery protocol and the entity-to-entity protocol(s) may use the shared memory to physically pass control information and data between the system unit and feature adapters.

The internal operation and distribution of the delivery service is logically structured into a delivery layer and a physical layer. The delivery layer supports the delivery of control blocks between "entity" pairs. The physical layer supports the delivery protocols used between "unit" pairs (delivery agents). The definition for the delivery protocol is based upon a Micro Channel form of physical connectivity between system units and/or feature adapters.

Users of the delivery service (entities representing clients and/or servers) are the next higher layer of service. Understanding the overall operation of the entity-to-entity layer services was key to defining the services to be provided by the underlying delivery service and the protocols needed to support the distribution of these services among the system units and feature adapters.

This layered structure for delivery services is shown in Figure 5.

This layering of delivery services allows the various entity-to-entity protocols to share a common delivery service. The delivery protocol can be full-duplex and free-running while individual entity-to-entity protocols may be half-duplex and of the master / slave form. The delivery protocol allows the delivery sup-

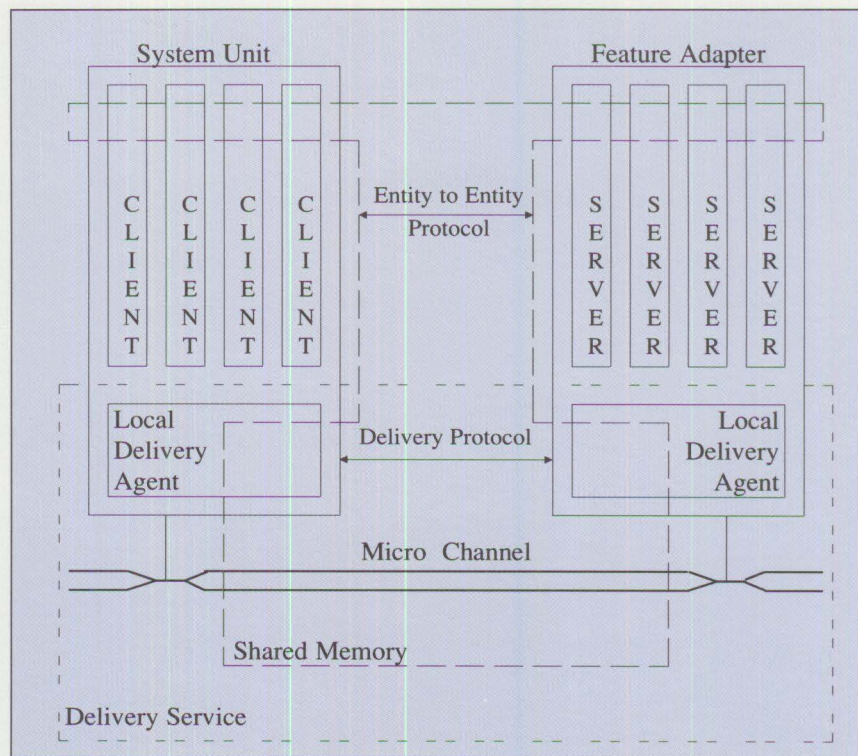


Figure 4. Generic Delivery Service

port to be mapped onto the different forms of the physical level protocols that must be used with different unit-to-unit pairings; that is, system unit-to-feature adapter, feature adapter-to-system unit, and feature adapter-to-feature adapter.

Delivery Service Facilities

The delivery service facility provides two operational modes: Locate mode and Move mode. The following describes the services, functions, and protocols provided by each and a brief description of the underlying control structures.

Locate Mode Support

The Locate mode form of control block delivery supports a control structure that has a relatively fixed

control block structure. The control blocks are delivered to the server one control block at a time. The Locate mode control block delivery structure is shown in Figure 6.

Locate mode control block delivery provides:

Multiple Devices per Adapter:

Subsystems will generally provide support for multiple devices and/or resources. These may be small computer system interface (SCSI) devices, LAN connections, X.25 virtual circuits, integrated-services digital network (ISDN) channels, communications lines, or processes. The delivery mechanism supports the delivery of requests to specific devices and/or resources through the use of device identification numbers (for example, Device 1, Device 2, Device n).

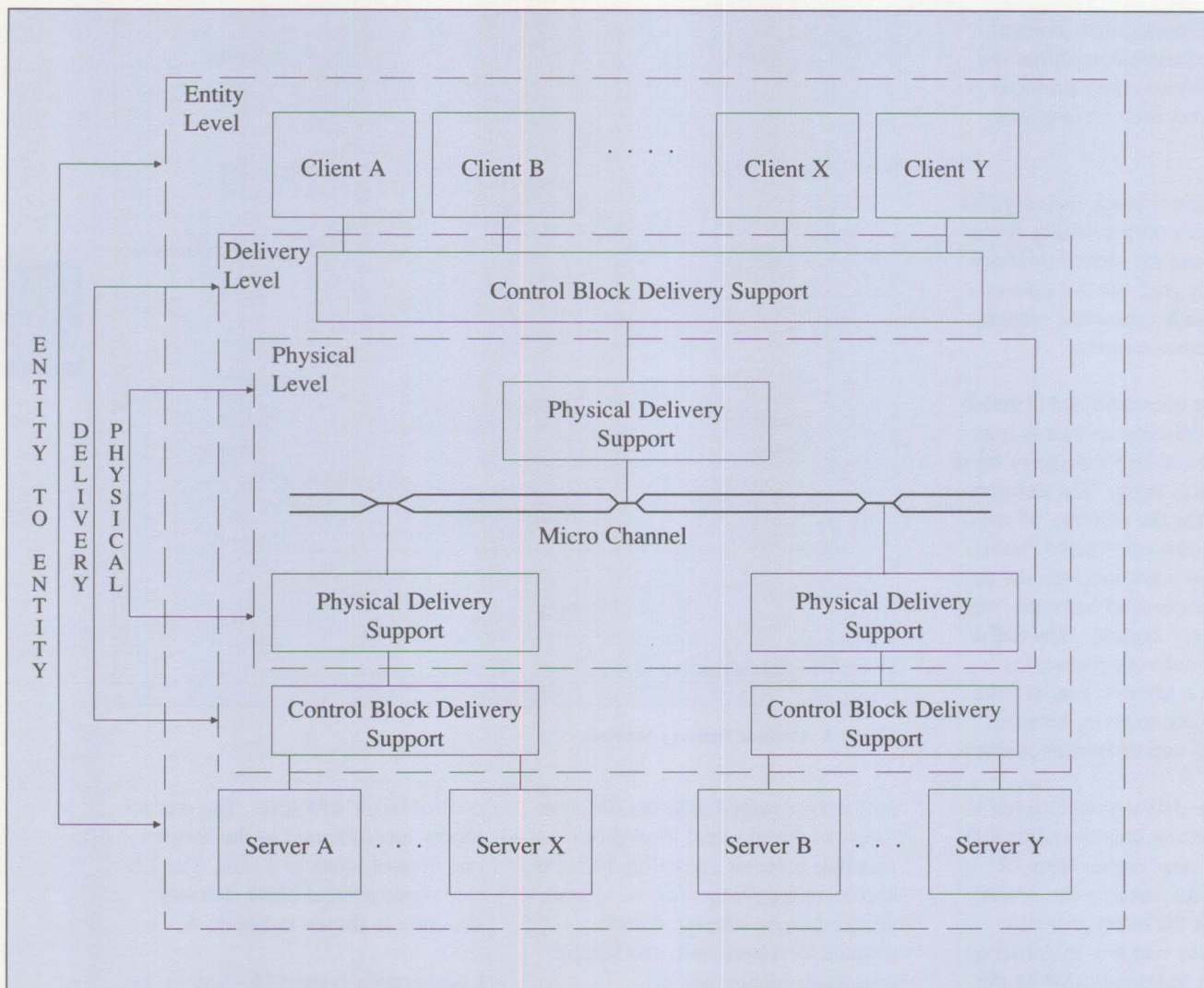


Figure 5. Delivery Service Structure

Subsystem Management: There is a requirement to deliver subsystem management information as well as control information to the subsystem. The subsystem manager is assigned Device identification number 0 and receives all subsystem management information.

Requests to Devices: To use a device or resource, a program (the client) sends requests in the form of control blocks to a specific device or resource (the server) and receives replies from the device or resource

for those requests. A single program or multiple programs operating in a system unit can be using a device or resource in a subsystem. These programs may be using the same or different devices or resources in the subsystem.

Command and Data Chaining and Detailed Status: The control structure defined for Locate mode provides for an immediate request, a request made up of multiple control blocks chained in a specific order and treated as one logical re-

quest, or using an Indirect List for chaining multiple buffers associated with a given control block.

Figure 6 shows a sample control request structure that consists of two control blocks (command chaining). The first control block uses an Indirect List to reference multiple buffers (data chaining). The other has a single buffer. The commands and parameters are contained within the individual control blocks.

The structure also provides for handling status information in case of exceptions during the processing of a request. The status is placed in a Termination Status Block. In order to handle termination at any point in a chain, a Termination Status Block is associated with each control block in the chain.

Use of Direct Memory Access

(DMA): The Locate mode form of delivery defines the interfaces to support the case where both the control structure for a request and the data associated with a control block are transferred between the system unit and the feature adapter using DMA operations managed from the subsystem.

Interrupts: The delivery service allows the builder of a request structure to define when and under what conditions interrupts should be generated. Generally there will be one interrupt per request. This will occur at the end of the request for exception-free operation. Additional interrupts may be requested in any control block in order to synchronize activities. Explicit commands are used to reset device interrupts.

The flow of the interrupt processing is from the hardware to the operating system kernel to the device driver (interrupt portion) and, when ABIOS is used, to the Advanced Basic Input / Output System (ABIOS) interrupt entry point. The interrupt processing uses information provided as part of the request/reply interface to determine which of the devices or resources in the subsystem caused the interrupt.

Locate Mode Control Areas

The architecture identifies the specific control areas in I/O space to be used, as well as the protocols for ini-

tializing and using a feature adapter in Locate mode.

Because multiple feature adapters of the same, or different types may be used in the system, the base address for the I/O space of each feature adapter must be defined during setup. The I/O control areas used by a feature adapter are shown in Figure 7 as offsets from the I/O base address.

In the following descriptions, the term "port" refers to a byte or set of contiguous bytes in the system.

Request Ports: There are four types of request ports associated with sending requests to a resource or device in Locate mode.

The first, the Command Interface Port, is used to pass either the 32-

bit address of a control block or the first control block in a chain to a subsystem in a feature adapter. It is also used to pass immediate commands. These immediate commands are typically device-directed and control-oriented.

The second is the Attention Port. It contains an attention code and a device identifier. The attention code is used to inform the subsystem in the feature adapter that the Command Interface Port contains either the address of a control block or an immediate command. The device identifier indicates which device or resource on the subsystem the request is directed to.

The sequence of sending a request involves writing to the Command and Attention Ports in that order.

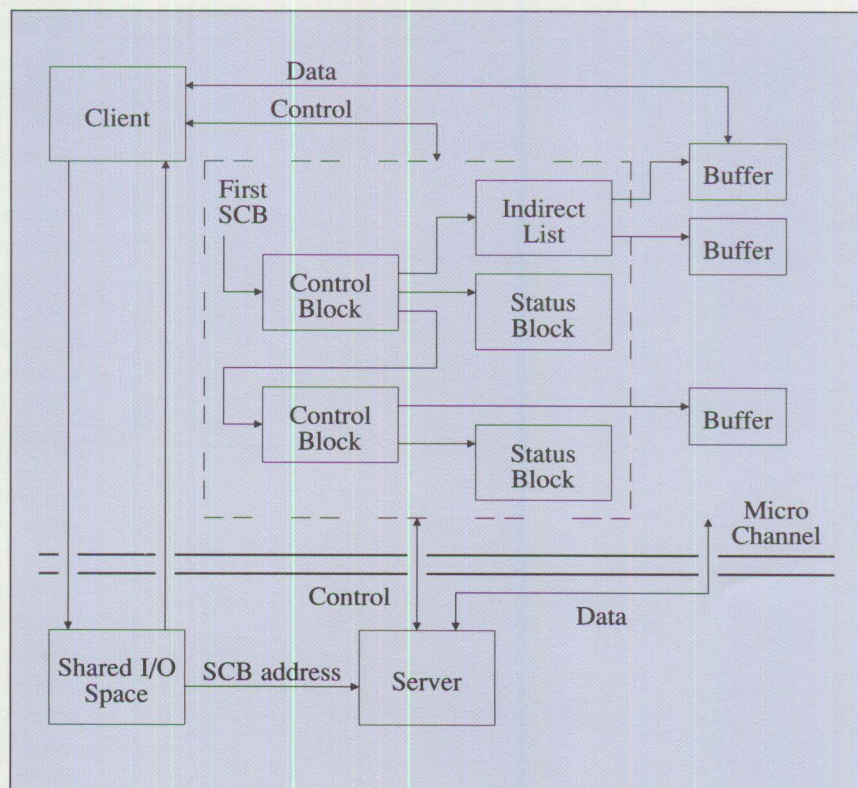


Figure 6. Locate Mode Control Block Delivery Structure

The third, the Interrupt Status Port, is used when the subsystem has completed processing a request or immediate command. It provides information needed by the system unit to associate a Micro Channel interrupt with a specific device on the subsystem. In addition to identifying the interrupting device, it also indicates whether or not an exception condition exists.

The fourth is the Command Busy/Status Port. It is used by the subsystem in a feature adapter to serialize access to the shared logic of the control block delivery service, the subsystem, or the device/resource. The port contains the following indicators:

- Busy - indicates that the subsystem is busy (using the shared logic). Commands submitted

while Busy are ignored by the subsystem in the feature adapter.

- Interrupt Valid - indicates that the contents of the Interrupt Status Port are valid and that the subsystem has requested an interrupt on behalf of one of its devices or resources.
- Reject - indicates that the subsystem has rejected a request (a Reset is needed to clear a Reject and allow the subsystem to resume accepting requests).
- Status - indicates the reason for the rejection.

Subsystem Control Port: The Subsystem Control Port is used to pass control indicators directly to a subsystem that cannot be easily handled by requests to subsystem

management. The port contains the following control indicators:

- Enable Interrupts - indicates that interrupts should be enabled or disabled for all devices attached to the subsystem in the feature adapter.
- Enable DMA - indicates that DMA operations should be enabled or disabled.
- Reset Reject - indicates that a reset of the reject state of the subsystem should be performed.
- Reset - indicates that a reset of the subsystem and all devices in the subsystem should be performed.

Device Interrupt Identifier Ports:

Interrupt status for all devices and resources associated with a subsystem are reported to the system unit through the Interrupt Status Port. However, when the optional Device Interrupt Identifier Port(s) are used, only the interrupt status for immediate commands will be reported through the Interrupt Status Port. All other interrupts will be reported using the Device Interrupt Status Port(s). When a device or resource completes processing a control block, it sets the bit in the Device Interrupt Identifier Port corresponding to its device or resource identifier, and then interrupts the system unit.

By using these optional ports, an interrupt handling program can process multiple control block interrupts on a single Micro Channel physical interrupt. This is accomplished by reading the Device Interrupt Identifier Port(s), and using the special immediate com-

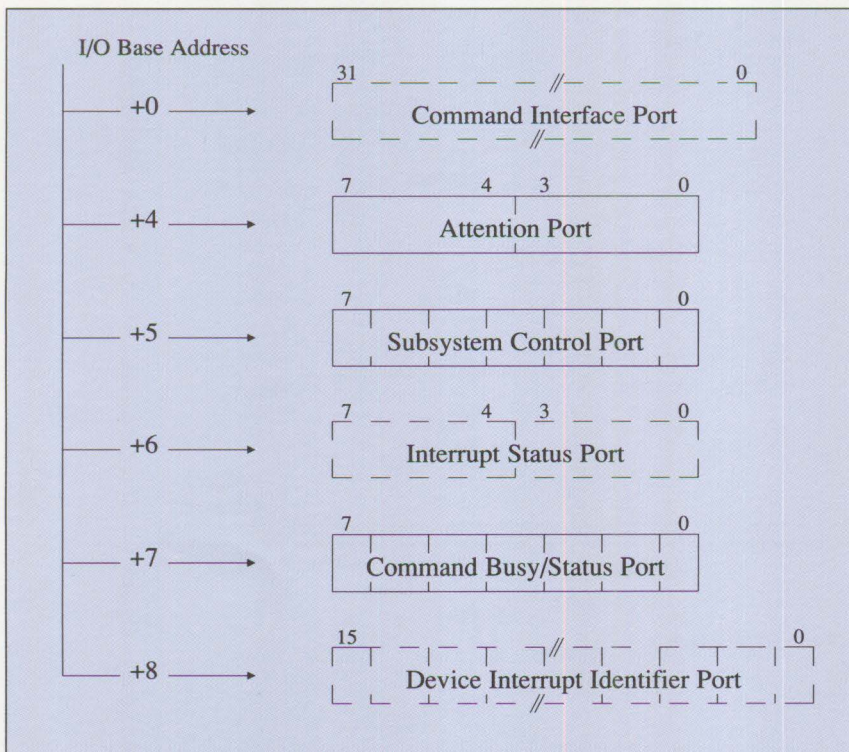


Figure 7. I/O Control Areas

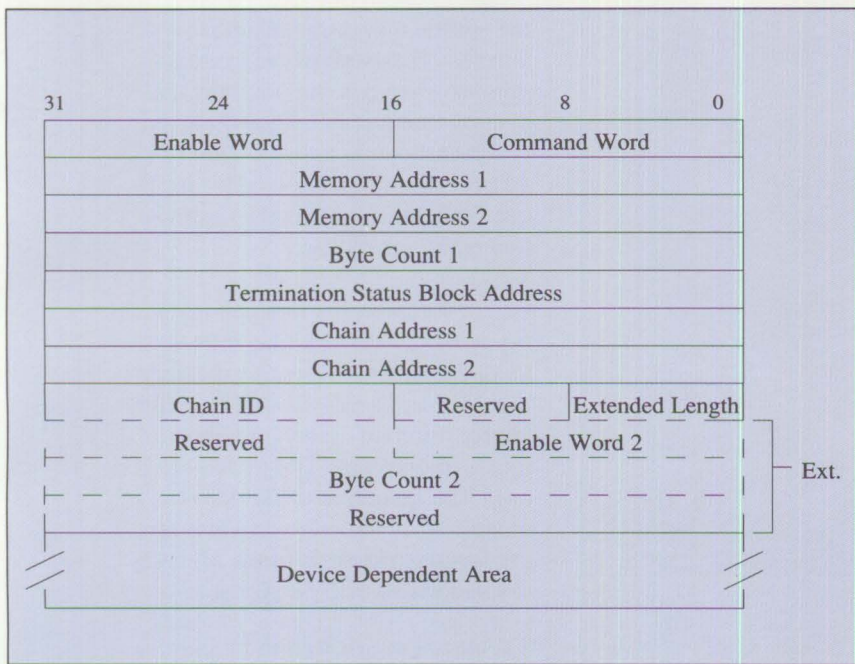


Figure 8. Control Block Format

mand, Reset Subsystem Control Block Interrupts, to clear the interrupt requests for the device.

Before issuing requests, the system-unit software must ensure that the subsystem is enabled to accept new requests, that is, not Busy or in the Reject state.

If virtual memory is being used, the system-unit software must ensure that all control blocks, Termination Status Blocks, Indirect Lists, and data areas associated with a request are locked into memory.

Control Blocks

The structure and content of a typical control block is shown in Figure 8.

There are two formats for the control block: basic and extended. Both forms share all of the fields shown in solid lines. The remaining fields (shown in dashed lines) are present only in the Extended

Format. The Device Dependent Area is also present in both forms. However, the actual location of the area within the control block is dependent upon whether the basic or the extended form is used.

The physical address of the control block must be placed in the Command Interface Port and the device address and attention code in the Attention Port, in that order.

Indirect Lists

An Indirect List is a variable-length list consisting of address-count pairs used to support data chaining. Both the address and the count are four bytes. The length of the list is contained in the control block that points to the list. The format of the Indirect List is shown in Figure 9.

Termination Status Blocks

In addition to the exception/no exception indication in the Interrupt Status Port, the SCB architecture provides for detail status information to be reported for each command. The status information is reported in the Termination Status Block (TSB). Each control block includes a TSB address to which a subsystem writes completion or termination status for that control block. The format of the basic TSB is shown in Figure 10.

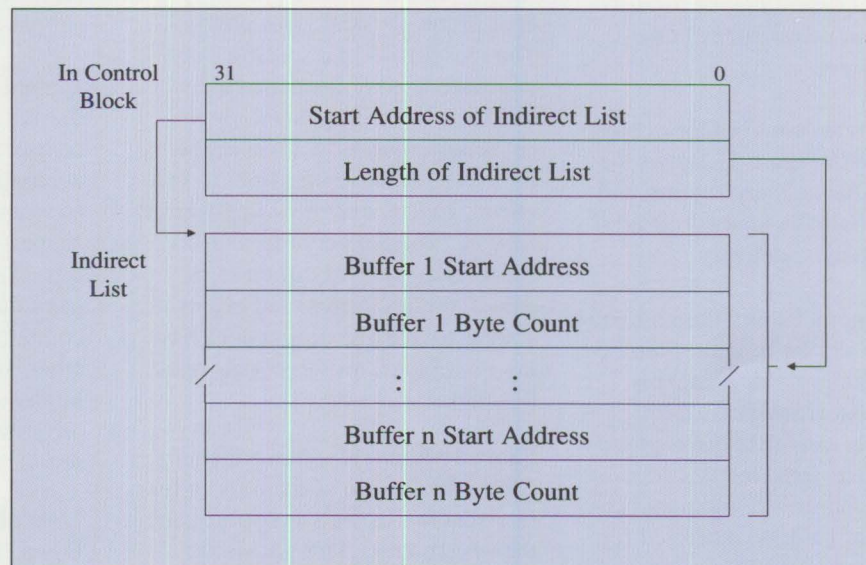


Figure 9. Indirect List Format

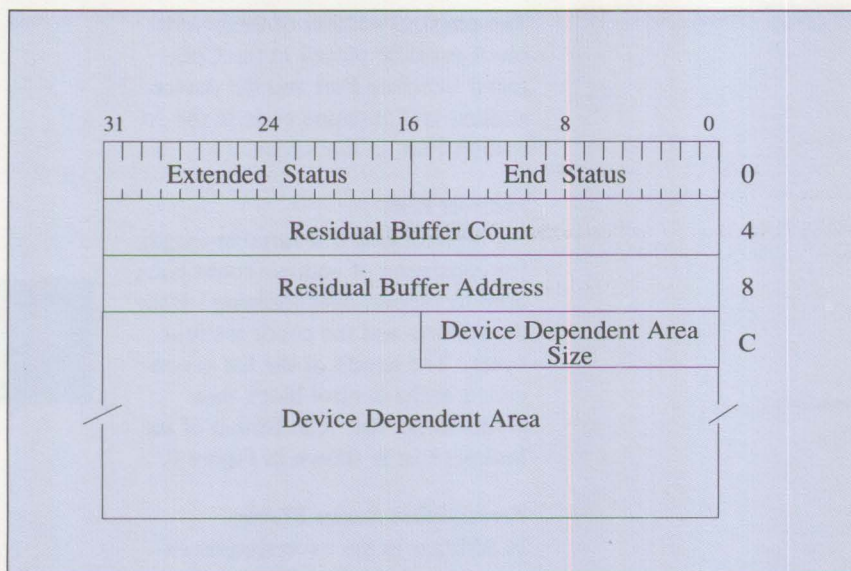


Figure 10. Termination Status Block Format

Move Mode Support

The Move mode form of the control block delivery supports a control structure designed to allow a variable-length list of control-element primitives to be used to deliver control information to a server. This variable-length list may contain requests, replies, error, or event notifications for a specific device or resource in a subsystem, or for different devices or resources in the same subsystem.

The Move mode control block delivery structure in Figure 11 shows the interface to the delivery support and the various memory spaces related to control element delivery.

The Move mode facility has an overall structure similar to that of the Locate mode facility; that is, clients build requests, requests are delivered to server, server builds replies, and replies are delivered to client. There are a number of additional capabilities that have been defined for the Move mode delivery facility.

The following are the additional Move mode capabilities.

Request/Reply Extensions: Request/reply extensions define support for error and event control elements in addition to request and reply control elements and the ability to have multiple outstanding requests between entity pairs. The flow of control elements is independent of the physical layer protocols. The two parties in a specific entity pair have been defined as the client and the server. In general, the client sends requests to a server and the server sends replies back to that client. Events may flow in either direction. The delivery mechanism supports the delivery of these requests, replies, errors, and events among source and destinations having multiple client server pair relationships.

Shared Memory: Shared memory allows for the use of memory in feature adapters as well as memory in the system unit. This allows the control structure used to convey control elements to be located in either

the system unit or the feature adapter. It also allows for various options when determining how the control structure is built and moved to the destination entity. (The server is the destination entity for requests and the client is the destination entity for replies.)

Figure 4 depicts the fact that memory is shared between the units and feature adapters. How this memory is shared and accessed will be different for different system environments (move/copy, DMA, and so on). The definition of the Move mode control structure provides for the fact that different forms of memory addressing will be needed.

Variable-Length Requests and Replies:

The Move mode support provides a control structure that allows a request to be made up of a set of variable-length control elements. At the entity interface, these elements are contiguous. This allows for chaining of control elements within a request to be done with a minimum of address manipulation. It also allows for a minimum-size control block for passing simple requests and replies.

Variable-length control elements addresses the need to have smaller control structures, to be able to pass a variable number of request parameters and to have a simple way to support a number of different subsets. Some client / server entity pairs may choose to use complex combinations of primitives while others use a simple set. They are all implemented from the same set of primitives using the same delivery services.

Lists of Requests and Replies:

Figure 11 shows a sample set of control elements flowing from a client to a server and another flowing

from a server to a client; both flow on the shared memory delivery service pipes.

The delivery pipes defined for the Move mode control element delivery service have the following attributes:

- Full duplex

A pipe for each direction of delivery between units. This allows for the delivery of control elements in one direction independent

of the delivery of control elements in the other direction.

The control structure provides correlation between requests and replies.

- Multiplexing of entity pairs

Each pipe may have control elements for multiple entity pairs in the same pipe. The control structure provides for source and destination identification in the control elements to allow a set of control elements for different en-

tity pairs to be delivered in the same pipe.

- Intermixing of requests and replies

The control structure supports a mixture of request and reply as well as other control element types in the same pipe.

- Continuously running

The control structure permits the delivery of control elements in a continuous flow. Mechanisms

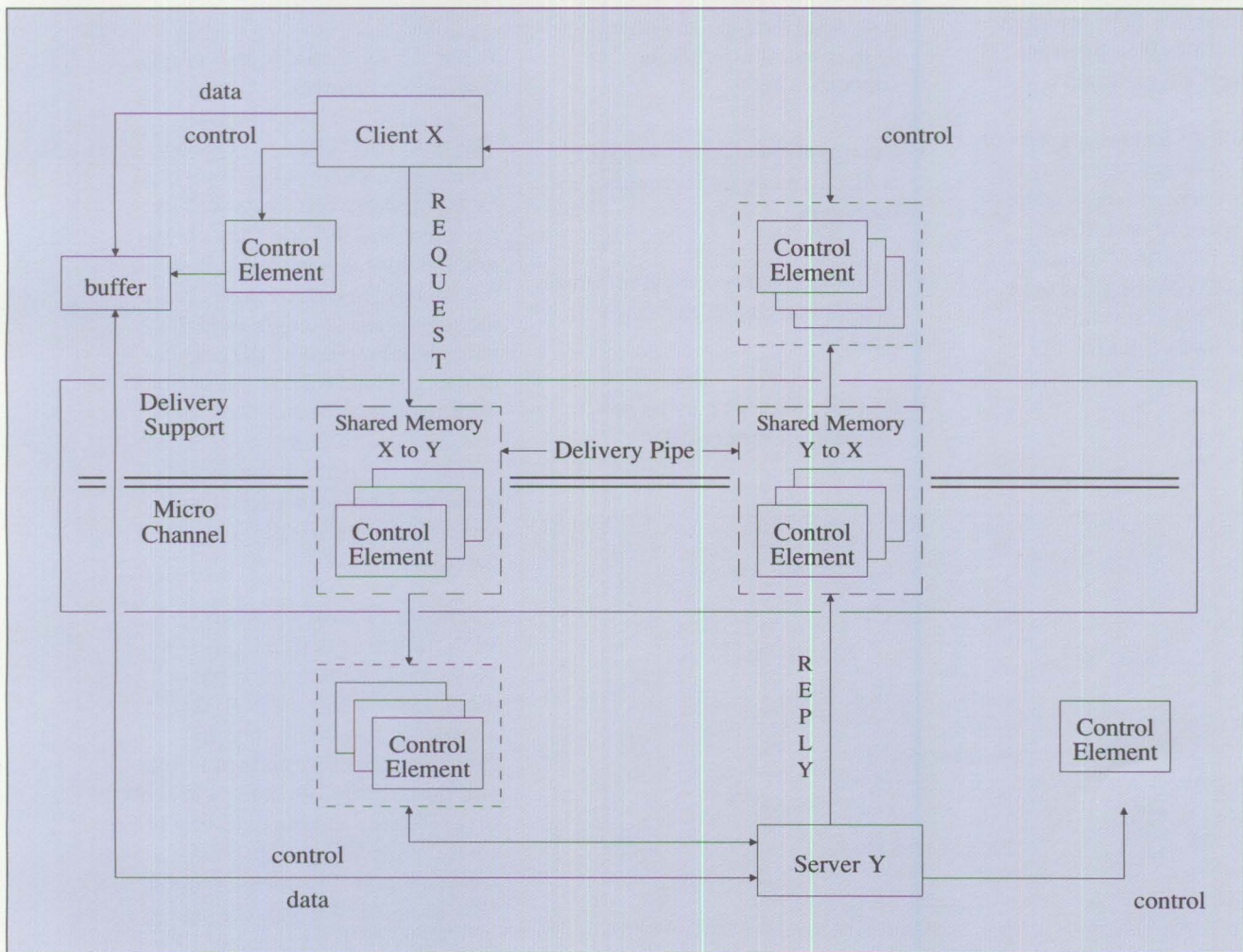


Figure 11. Move Mode Control Block Delivery Structure

are defined to provide a common way for entities to suspend the delivery of control elements at the entity-to-entity level, to notify the destination entity that a control element(s) is available and to provide for synchronization between the entities in the source and the destination.

Move Mode Control Areas

The architecture identifies the specific control areas in I/O space to be used as well as the protocols for initializing and using a feature adapter in Move mode. The control areas are essentially the same as those used for Locate mode, except the Command Interface Port, Interrupt Status Port, and Device Interrupt Identifier Ports are not used.

Because multiple feature adapters of the same or different types may be used in the system, the base address for the I/O space of each feature adapter must be defined during setup. The I/O control areas used by a feature adapter are shown in Figure 7 as offsets from the I/O base address.

Control Elements

Control elements are like control blocks. They are used to exchange control information between a client and a server. However, control elements differ from control blocks in the following ways:

- Control elements are variable in length.
- Control elements are self-describing.
- Control elements provide a means of specifying the destination, identifying the source, and indicating the type and urgency of the control information they contain.
- Control elements may optionally contain data as well as control information.
- Control elements contain information for correlating requests with replies.
- Control elements may be processed asynchronously.

To draw an analogy, a control element is like an envelope with a see-through window, while a control block is more like a post card. Both have a purpose and a use.

Delivery services use information in the window to deliver control elements, without knowing or understanding what is contained within the body of the control element.

Clients and servers use information in the window to specify the destination, identify the source, indicate the type and urgency of the control information, and correlate replies with previous requests.

Figure 12 shows the format of a typical control element.

The type, length, source, destination, and correlation fields are used by the delivery service as well as the client and server. They constitute the information visible through the window in the envelope. The remaining variable-length field, the value field, represents the contents of the envelope (that is, the control information). The structure, content, and length of this field is determined by the particular protocol being used between a client and server and is meaningful only to them.

Queueing Control Elements

With the use of delivery pipes in Move mode (which are implemented as circular queues), there is a tendency to want to mix the queueing capability defined to support the delivery of control elements between units and the queueing of control elements to a specific server. Figure 13 shows a request flow example of a delivery pipe as defined by SCB architecture. The delivery protocols support a free-flowing pipe between entities. Each

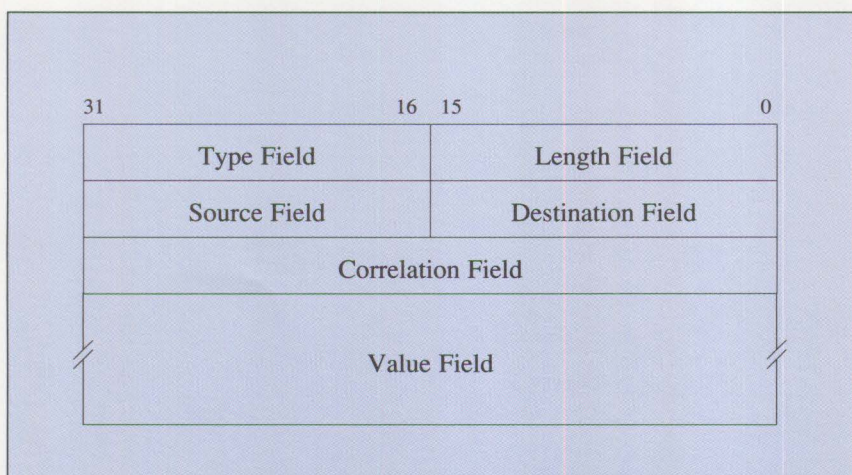


Figure 12. Control Element Format

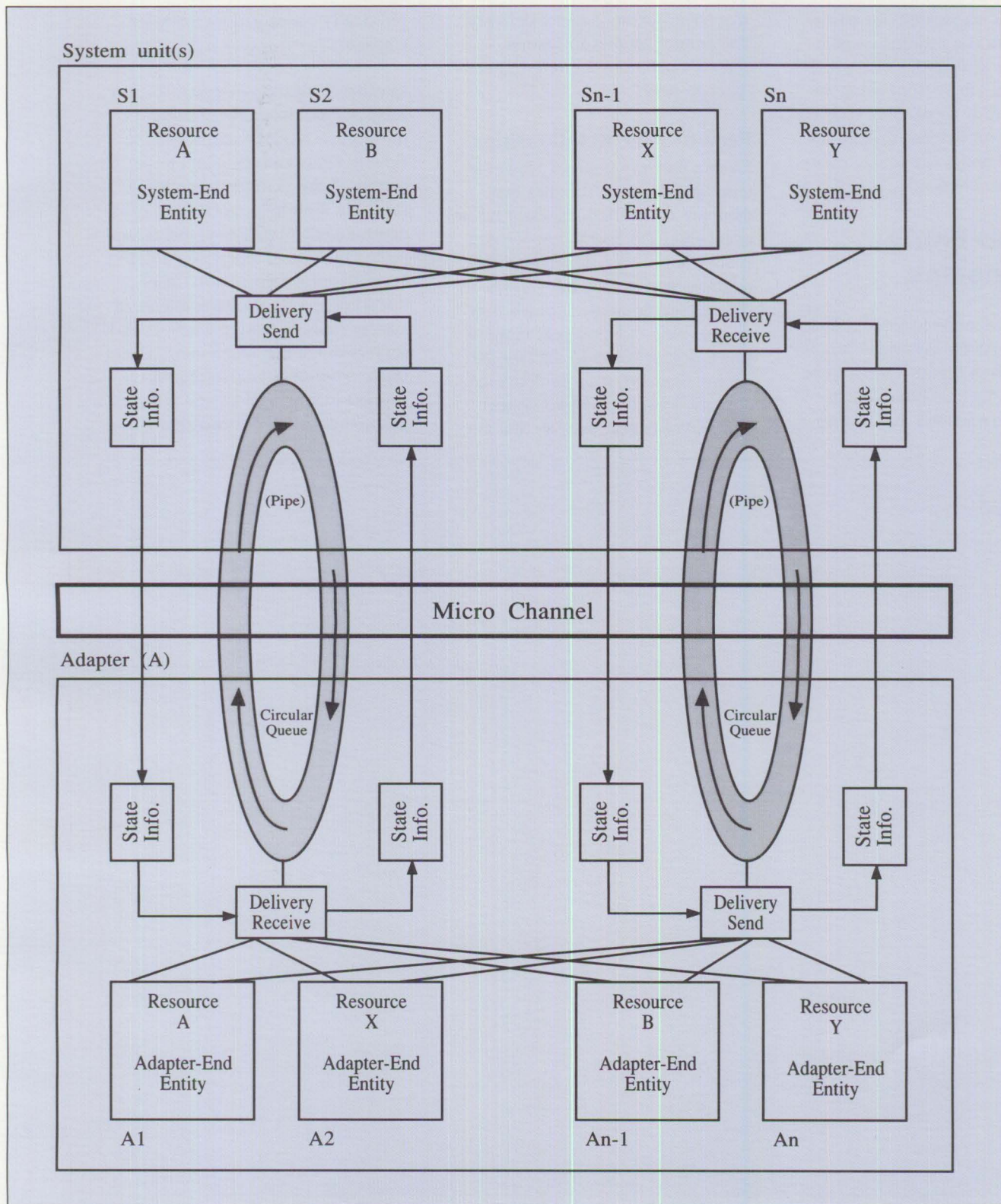


Figure 13. Handling Control Elements in Move Mode

entity pair is responsible for ensuring that there is a pending receive for elements sent so that one entity cannot block others from using the pipe. If there is no pending receive, the delivery service can discard the element and notify the source entity (sender of the element).

Entity-to-Entity Relationships

For the Move mode form of control block delivery, the generic configuration shown in Figure 4 must be expanded to include both system unit-to-feature adapter and feature

adapter-to-feature adapter delivery. The feature adapter-to-feature adapter operations are referred to as "peer-to-peer."

Peer-to-Peer Relationships

From a delivery point of view, the term "peer-to-peer" implies that there are no restrictions about where clients and servers may be located. It does not say anything about the relationship between the various client and server entities (which may be operating in a non-peer relationship). It also refers to the fact that control elements may be delivered directly between any two system unit and/or feature adapter that are

physically connected by the Micro Channel.

In order to operate in a peer-to-peer relationship, the delivery support must allow requests and replies to flow in either direction and to be mixed on the same delivery-level flow. In the Move mode form of control element delivery, this is supported by having independent delivery of control elements in either direction and by allowing clients in system units or feature adapters and servers in feature adapters or system units. Peer-to-peer also requires support in both system and feature adapters to resolve contention when

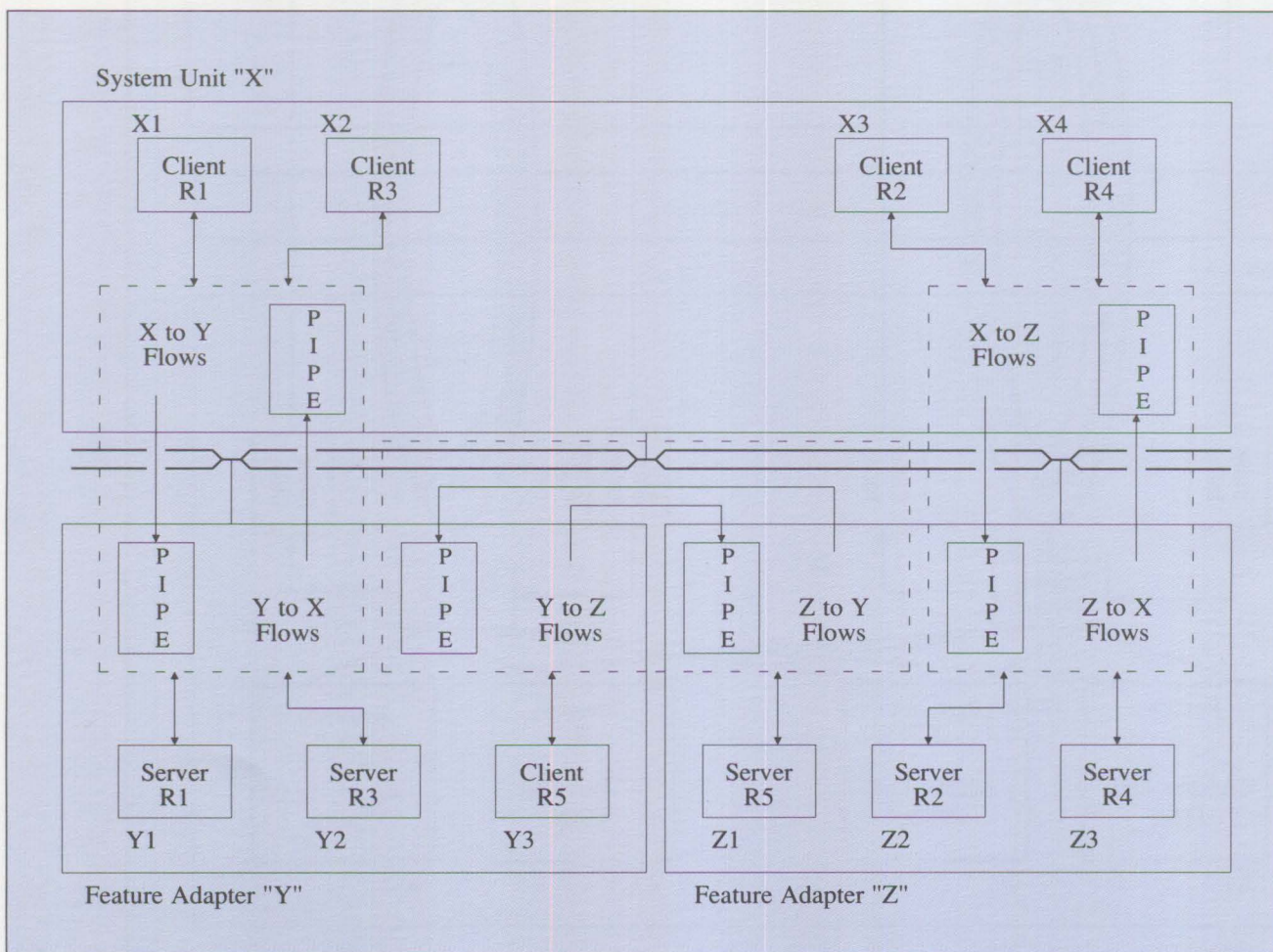


Figure 14. Peer-to-Peer Delivery Model

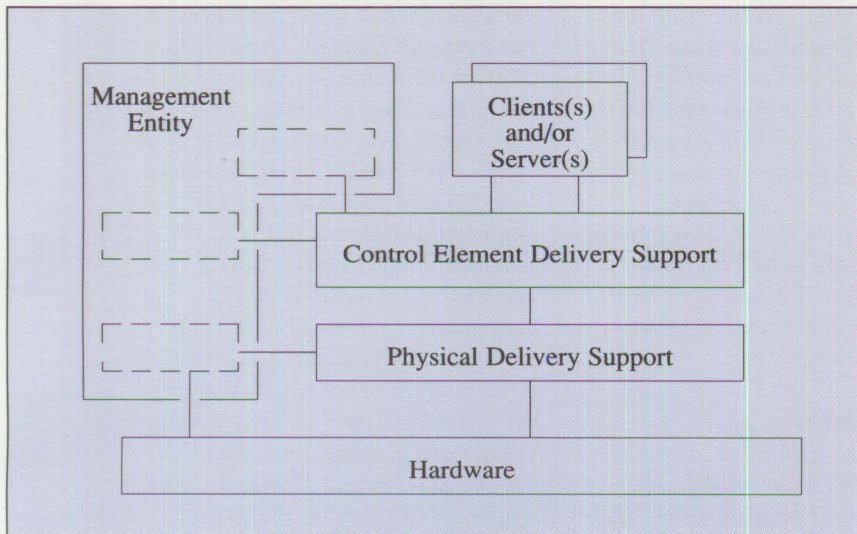


Figure 15. Delivery Management Services

two or more system units or feature adapters attempt to deliver control elements to the same destination at the same time.

The term "peer-to-peer" must be qualified by the level of support being discussed and not used as a unit-to-unit term without qualification.

Peer-to-peer delivery support has a pair of delivery pipes for each unit-to-unit pair with entities that communicate with each other. An example of this is shown in Figure 14. The labels R1, R2, and so on, on the client / server entities indicate the entity pairings. The dotted areas indicate the delivery support portion in each unit.

Figure 14 shows each delivery pipe as being distributed to both the source and destination. The definitions for the Move mode form of control block delivery are for cases where the actual physical queues implementing the pipes are in either the source or the destination, but not both.

Management Services

In addition to the peer-to-peer control element support, there is also a requirement to have a management structure to allow for the handling of various management services related to the operation of the delivery services.

The SCB architecture requires an entity in each system unit or feature adapter to be used for management support. This management entity,

which is shared by all the other entities, has an entity identifier of zero. The management entity structure is shown in Figure 15.

These management entities may use the delivery services to send control elements to or receive control elements from a system-level management entity. The system management entity, which does not have an entity ID of zero, provides system-wide management services (Figure 16).

This management structure may support management services of various types and is not unique to the delivery service. This means that it could be used for entity layer reporting and testing as well as delivery layer(s) management.

The management structure represents a unique form of client server support. The management services for a system are hierarchical even though the delivery support for clients and servers is peer-to-peer. This is an example of the fact that the basic delivery relationship is separate from the entity-to-entity relationship.

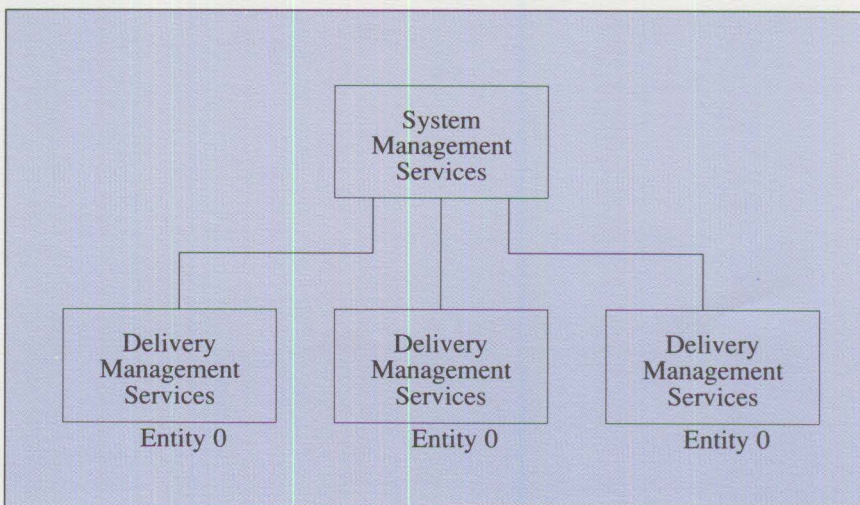


Figure 16. System Management Services Structure

Summary

The SCB architecture was developed to standardize the programming task of supporting Micro Channel bus master feature adapters, as well as the engineering task of designing the bus master programming interface. It gives the adapter designer the freedom to define the meaning and content of the protocols that the feature adapter supports, while at the same time providing the programs using these feature adapters with standards for important interfaces.

ABOUT THE AUTHORS

Frank M. Bonevento is a Program Manager with IBM. He joined IBM in 1964 and has held a number of technical and management positions in the areas of Systems Architecture, and in the design and development of systems software. He was a member of the design team that developed PL/S, a systems programming language widely used in IBM. Later Frank was a Programming Development Manager for systems software for

the IBM Series/1. He received an Outstanding Innovation Award for the design and implementation of the Series/1 PL/1 software products, and a Division Excellence Award for work on Advanced Systems Architecture for the Personal System/2. Frank has had inventions published in the IBM Technical Disclosure Bulletin and has patent applications in process on several others.

Ernie Mandese is an Advisory Engineer with IBM's Personal Systems Architecture group. He has been with IBM since 1979. He holds a bachelor of science degree in electrical engineering from University of South Florida. Ernie participated in the development of the IBM Personal Computer and the Personal System/2. He is co-author in the development of the Base Subsystem Control Block Architecture. He has several patents pending.

Joseph P. McGovern is an Advisory Programmer at IBM's Entry Systems Division development laboratory where he is involved in

the development of architectures for the Personal Systems. Joe joined IBM in 1984 after spending 15 years with Sperry Univac, where he was responsible for the development of the Distributed Communications Architecture (DCA) and Office Information Systems (OIS) architecture. He was a charter member of the ANSI/SPARC Distributed Systems Study Group and later served as the Chairman of ANSI and ISO/TC97/SC16 working groups responsible for developing the service protocol and definitions for the Session and Transport layers of the ISO Reference Model for Open Systems Interconnection.

Eugene M. Thomas is a Senior Technical Staff Member currently responsible for developing architectures for Personal Systems. Gene joined IBM in 1957 and has been involved in numerous design and architecture activities related to graphics, communications and distributed systems. These activities include work on early definition of software support for 2250 and 3270, and the definition of SNA.

Design Alternatives with Micro Channel Systems

Chet Heath
IBM Corporation
Boca Raton, Florida

The developer's choice of attachment design will affect the cost, performance, power consumption, and reliability of peripheral device adapters, and may impact other elements of the system. While the personal computer was a single-tasking system design, and only one adapter and device were typically active at one time, Micro Channel computer systems can additionally support simultaneous activity of multiple tasks and/or users. This implies that many adapters and devices can now be active at the same time. The concept is called I/O concurrency and is a major new element of design for these systems.

All of the simpler personal computer design types are retained in Micro Channel systems in order to allow support of logically identical adapter designs for PC applications. Five times as many direct memory access (DMA) adapters can be supported on the Micro Channel standard as are supported on the IBM Personal Computer. A powerful new type of adapter, the bus master, is defined to extend the capabilities of systems beyond the limitations of a single processor that controls all I/O.

Primitive Designs

Personal computer designs typically depended heavily on the support of the processor for even the most rudi-

mentary operations. This was permissible, because the processor was dedicated solely to one adapter function at a time. The most cost-effective approach was to implement the minimum of logic and depend on complex and lengthy support software to operate the device. Adapter examples of this type are:

- Continuous polled
- Interrupt per character
- Memory mapped

In general, these adapter types can induce limitations in the system. They consume processor time, at the expense of application processing, and require that the system processor be in control when I/O devices operate. Depending on the degree to which the I/O devices are active, such adapters can also degrade system performance in a multitasking or multiuser system. When other masters and DMA adapters compete for bus ownership, significant delays can occur when the processor must first gain

control of the channel before operating the adapter. Yet these adapters are often implemented in a system because they provide register-level compatibility to existing personal computer (DOS) applications.

Continuous Polled: In a continuous polled adapter, such as the game port, the processor periodically requests information, or "polls" adapter registers for status of the attached device. Repetitive polling can consume much, if not all, of the computer's processing power. Such adapter designs are best suited to the only purpose for which they are used today – playing computer games.

Interrupt per Character: Interrupt-per-character designs are a step better than polling; the processor is free to support other duties until an interrupt request for processor attention is placed on the channel. The interrupt request to move a character of data between the adapter and storage, can consume 200 or more processor instructions just to change the processor's attention. This overhead may be acceptable in support



of low performance devices (such as a keyboard adapter) where the requests are infrequent and represent only a small percentage of the total capability of the CPU. Faster devices, such as high speed communications ports and printer interfaces, can easily consume all of the processor's power as the rate of interrupts and the cumulative overhead increases.

Memory Mapped: A memory-mapped adapter shares a segment of the storage area in the system with the processor. Large blocks of several thousand characters can be written to the mapped-memory area without disturbing the processor. An interrupt can signal the processor that the block is now available to the processor and the interrupt "overhead" is thereby "amortized" over many transfers. To some degree, this design type can be more efficient than most interrupt-per-character designs.

Unfortunately, there are other problems. The shared memory is as-

signed to a fixed address in an area that is reserved for input / output operations. The data often must be copied to or from an area of memory, dynamically defined by the operating system, for the storage of program data. The copy operation takes time and can negate much of the performance advantage. While the two memories can be controlled in a similar way by the processor, the shared memory is not physically part of the general memory system. It is implemented on the adapter card, adds cost to the card, takes up space and consumes power. The arbitration for access to the memory by the system can further diminish performance by adding time to every data transfer operation. Furthermore, there are a limited number of the fixed assignments of shared I/O memory. Consequently the number of configurations possible with other designs may be limited as well.

Advanced Designs

Two important design alternatives that do not depend on the processor are now available to the designer: DMA designs and bus masters. The number of direct memory access designs was limited in PC, PC XT, and AT computers due to prior fixed and permanent assignment of the DMA data transfer channels. With Micro Channel architecture, as many as 15 DMA adapters can now be supported. Bus masters – adapters that can directly control memory and other adapters in the system – provide the greatest freedom in application design, are the most efficient with system resources, and allow the minimum dependence on the system processor. As many as 15 bus masters can be concurrently installed and active on the Micro Channel standard interface. While primitive adapters are limited by the instruction processing power of the system processor, the advanced types are typically limited by the capacity of the channel to transfer data, called throughput; the through-



put of the Micro Channel is very high indeed!

DMA Designs: A DMA design can free the processor from responsibility to transfer data, and is an ideal choice for low cost, medium performance adapters that transfer records of a few thousand bytes or less at a time. The adapter requests control of the bus, and one DMA channel which has been assigned to support the adapter then moves the data to or from system storage. Such adapters may implement the burst capability of Micro Channel architecture to move large blocks of data directly to the memory area defined by the operating system for storage of program data.

The DMA design does not require the shared memory of memory-mapped designs, may be less expensive, transfers data directly to the program data storage area, does not depend on processor activity, and still amortizes the interrupt overhead over a large number of transfers. The DMA adapter is, however, limited to sequential organization of data in storage. Operating systems may not always be capable of reserving long blocks of contiguous system memory for temporary storage of program data; consequently DMA adapters are best suited to records shorter than approximately four thousand characters (the page size for 386 and 486 processors).

Bus Master Designs: Bus masters provide all of the capabilities of DMA designs plus two additional, powerful advantages:

- Increased data transfer efficiency
- Random memory or I/O selection

A DMA adapter may require as many as four operations on the Micro Channel interface to transfer data with a data destination; a bus master will require only one operation. Even more important, a bus master can randomly address memory and I/O devices and is, therefore, not limited to sequential data transfer. It can intermix operations in support of a number of devices, or a number of data storage areas. Bus masters are optimal for the attachment of concurrent processors, intelligent subsystems, or any adapter that transfers either long lengths of data and/or high-speed data.

The ability of a bus master design to directly control and randomly read and write storage is an enabling characteristic for a Subsystem Control Block (SCB) architecture. With SCB architecture, the processor places formatted blocks of control information in storage where a bus master can access them. These blocks specify the location of data buffers in storage, the length of the data, what command an adapter should pass to the controlled device, and perhaps the location of additional control blocks that can be chained together to create complex macro operations. Subsystem control blocks complete the other half of the equation in distributed multiprocessor design. They define the protocol and control operations for bus masters to work together and under the control of an operating system.

The bus master, however, exacts one tax. Often the power, cost and design complexity are increased for this type of design. The tax is usually justified for attachment of collections of I/O devices, such as in a small computer system interface (SCSI) device controller. A bus

master may also be cost-effective for the connection of expensive and perhaps complex functions, such as a processor, local area network, fixed disk, or display adapter.

The developer of products for Micro Channel systems is presented with a number of enhanced and new choices for the attachment of peripheral devices. Within each type of design, there are subdivisions and refinements that further extend the design options. The broad range of choices makes it possible for Micro Channel systems to be designed for simple single-tasking situations, for advanced microsystems, and even for minicomputer or mainframe applications. Having a broad range of applications is the greatest advantage of all to the developer; it ensures the greatest possible market opportunity for the design.

ABOUT THE AUTHOR

Chet Heath is a senior engineer at IBM's Entry Systems Division laboratory in his twentieth year with IBM. He holds a bachelor of science degree in electrical engineering from the New Jersey Institute of Technology and a master of science degree in electrical engineering from the IBM LSI Institute at the University of Vermont. He refers to himself as "the oldest living survivor of Micro Channel architecture" because he was involved in its definition since inception and gave the architecture its name. He has received IBM Quality, Outstanding Technical Achievement, Outstanding Innovation, and Corporate Technical Achievement Awards. Chet also has attained the eighth level of invention awards. He is presently assigned to development of Micro Channel strategic enhancements.

The IBM PS/2 Micro Channel SCSI Adapters

Andrew McNeill
IBM Corporation
Boca Raton, Florida

This article describes the features and operation of the IBM-developed SCSI adapters for PS/2 Micro Channel computers. These adapters allow peripheral devices that conform to the Small Computer System Interface (SCSI) standard, such as high-performance fixed disks, optical disks and printers, to be connected to IBM PS/2 computers.

The SCSI-1 standard was approved by the American National Standards Institute (ANSI) in 1986. This standard, which defines an electrical interface (bus) and communications protocol, supports the interconnection of up to eight SCSI-compatible devices. Usually one of the devices is a SCSI adapter installed in the computer system; the other SCSI de-

vices are peripherals, such as fixed disks, optical disks, laser printers, tape drives and scanners. A diagram of a PS/2 Micro Channel system with a SCSI subsystem installed is shown in Figure 1. The subsystem consists of a SCSI adapter, cable(s), and one or more SCSI devices.

Two SCSI adapters have been developed by IBM. Both adapters are PS/2 Micro Channel-compatible bus masters that support the connection of up to seven SCSI peripheral devices. Up to four SCSI adapters can be installed in a single PS/2 Micro Channel system, providing the capability to attach up to 28 SCSI devices to a single PS/2 computer.

Both adapters support data transfer rates of up to 5 megabytes per second across the 8-bit, parallel data path of the SCSI bus. The higher-performance adapter is a 32-bit Micro Channel bus master with a built-in, 512 KB disk cache. The lower-cost adapter is a 16-bit Micro Channel bus master. The 32-bit

adapter supports data transfer rates of up to 16.7 megabytes per second between the adapter and PS/2 system memory across the Micro Channel. Each adapter has both an internal connector for cabling to SCSI devices located inside the PS/2 system unit, and an external connector for cabling to SCSI devices located outside the system unit. Up to 20 feet (6 meters) of combined internal and external cable length is allowed for each SCSI adapter. This cable-length limit is due to the the high-speed nature of the bus signals. For details of SCSI device and cable installation, refer to the section in this article titled "SCSI Subsystem Hardware and Software Configuration" and the installation instructions for specific adapters and devices.

Figure 2 illustrates a PS/2 computer with a SCSI subsystem. A SCSI adapter and SCSI fixed disk are connected inside the system unit, and a SCSI Compact Disk Read-Only Memory (CD-ROM) drive is connected by a cable to the external connector on the adapter. The CD-ROM drive is housed in an external expansion unit that contains a power supply for the CD-ROM drive electronics and the cable connectors.

A CD-ROM drive can be used to access over 500 megabytes of data (for example, text, graphical images, audio and video information). Most CD-ROM drives accept standard audio CDs also.

Figure 3 shows a maximum configuration of 28 SCSI devices connected to four SCSI adapters installed in a single PS/2 system unit. Up to 28 SCSI devices can be attached to a PS/2 computer to support the storage of large volumes of digital infor-

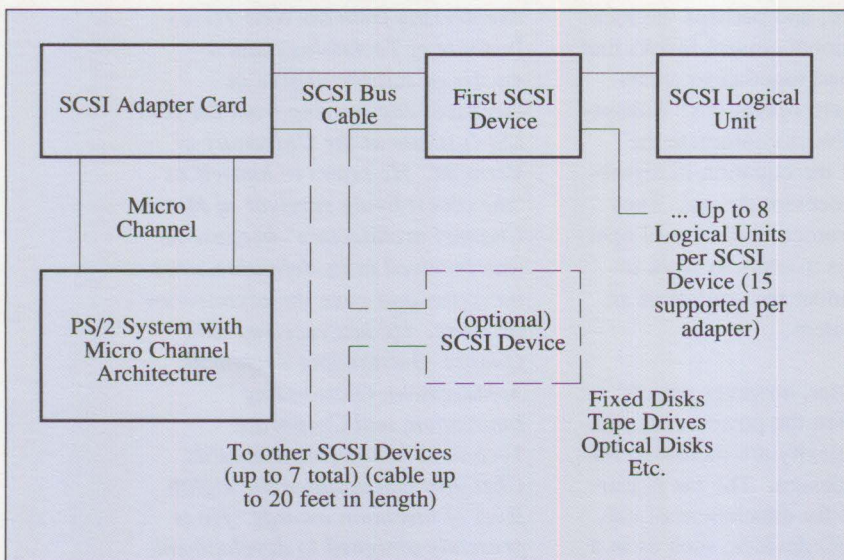


Figure 1. SCSI-Based PS/2 System

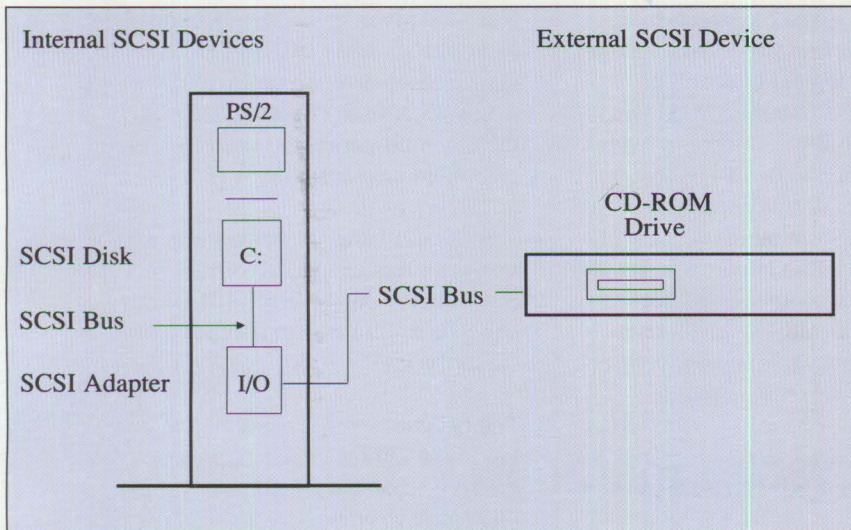


Figure 2. Typical PS/2 with SCSI Adapter and Devices

mation for large data bases, optical-image libraries, and other custom applications.

Support can be extended even beyond those 28 devices. The SCSI standard allows for each SCSI device to optionally support several "logical-unit numbers." By using these devices with logical-unit support, up to 60 SCSI devices can be connected and operated concurrently by a single PS/2 computer. The Micro Channel bus master capability of the IBM SCSI adapters allows each adapter to operate up to 15 SCSI logical units concurrently without involving the PS/2 system microprocessor. This allows a multitasking operating system such as OS/2 to provide more processing power to support multiple applica-

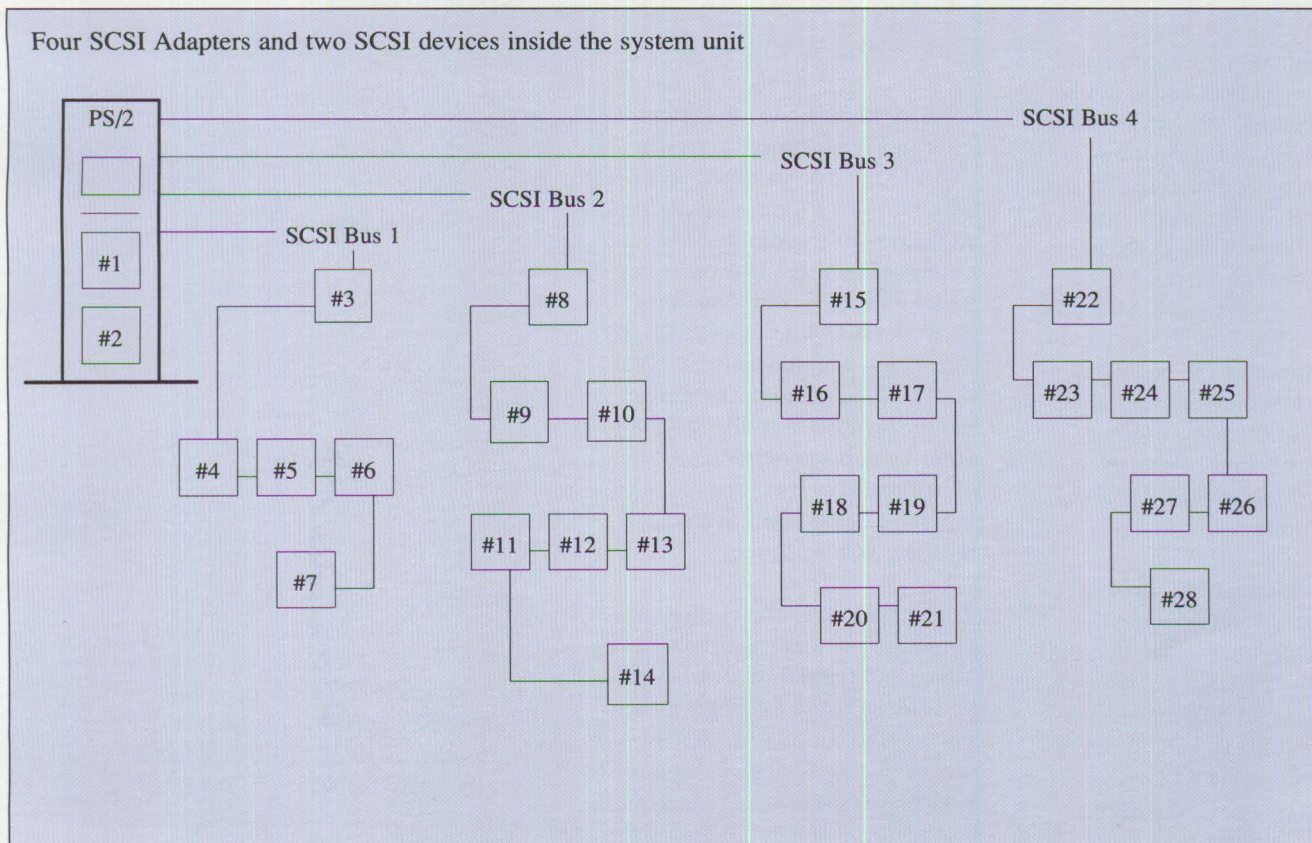


Figure 3. SCSI-Based System with 28 SCSI Devices Attached

tion programs running concurrently, because the IBM SCSI adapters manage the bus communications tasks without involving the system processor in SCSI subsystem control.

Note: For the remainder of this article, adapter information refers to both SCSI adapters unless noted otherwise. The 32-bit cached adapter is software-compatible with the 16-bit adapter, and the disk-cache function is transparent to the operating system and application software.

SCSI Adapter Features

When installed in a PS/2 computer, an IBM SCSI adapter is actually a "computer within a computer." Having an on-card microprocessor and custom control program (microcode) contained in a read-only-memory (ROM) chip, the adapter is a dedicated-function microcomputer system on an adapter card. Using custom integrated-circuit chips that provide the Micro Channel bus master interface and other chips for data handling and SCSI interface functions, the adapter microcode performs operations as requested by the system software. The adapter's bus master capability allows the on-card microprocessor to control the transfer of data into and out of the PS/2's system memory without using the system processor or system-board, direct memory access (DMA) controller.

The adapter controls the flow of data to and from several possible sources and destinations within the system. The PS/2 system memory, SCSI devices and SCSI adapter memory are three possible sources or destinations of data. During execution of a command, the adapter may perform a number of different data transfers between these entities.

Each SCSI adapter supports overlapped command processing for up to 15 SCSI logical units, all of which share a common SCSI bus. The devices allow command overlapping by logically disconnecting from the bus during times when no data transfers are taking place with the adapter. This allows commands to be passed to other SCSI devices while devices already given commands are busy performing their re-

*An IBM SCSI adapter
is actually a
"computer within
a computer."*

spective commands. For example, after receiving a Read command, a SCSI fixed disk usually disconnects while it moves the read/write heads to the proper track and locates the requested sectors. If another command request is sent to another SCSI device while the SCSI fixed disk is logically disconnected, the adapter sends the new command to the proper device so that both devices can operate simultaneously. When ready to resume data transfer, the SCSI devices will reconnect with the SCSI adapter.

The adapter also allows chaining of commands to a particular SCSI device, resulting in improved system performance and ease of programming. Another adapter feature, the scatter/gather capability, supports data transfers between the adapter and up to 16 non-contiguous areas of system memory. This capability enables virtual-mode operating sys-

tems and other special applications to further increase system performance by combining up to 16 commands into a single command, thereby eliminating the need to execute them one at a time.

The SCSI adapter described implements the Locate Mode form of the Subsystem Control Block (SCB) architecture described in another article in this publication.

The SCSI adapter is described in detail in the *IBM Personal System/2 Micro Channel SCSI Adapter Technical Reference* and the *IBM Personal System/2 Micro Channel SCSI Adapter with Cache Technical Reference*.

SCSI Basic Input/Output System (BIOS)

In addition to the microcode ROM chip located on the adapter card, there are two other ROM chips, which contain the SCSI adapter BIOS and BIOS Power-On Self-Test (POST). These software routines are executed by the PS/2 system processor to initialize and test the SCSI subsystem after a system reset and to provide a low-level, adapter-hardware-independent programming interface. The interface allows higher levels of software executing on the system processor to control the subsystem. The POST supports automatic configuration of subsystem functions and performs a configuration check after a system reset. The POST also ensures that all fixed disks attached to SCSI adapters in the system are made ready for operation. The POST supports coexistence with other types of fixed-disk adapters that may be installed in the same system. If more than one SCSI adapter is installed in a system, only one adapter's BIOS is re-

quired to support all SCSI adapters installed in the system.

BIOS software on the adapter consists of two primary interfaces. The BIOS interface supports SCSI device and adapter functions for the DOS environment. The Advanced BIOS (ABIOS) interface supports operations for the OS/2 environment. Other operating system environments may require specific device drivers for the SCSI adapters.

For compatibility with existing versions of DOS and OS/2, the SCSI BIOS provides a standard software interface for SCSI fixed disks and supports the operation of both SCSI and non-SCSI fixed disks within the same PS/2 computer.

New BIOS and ABIOS functions have also been included in the SCSI adapter ROM BIOS to support operation of SCSI devices other than fixed disks. These types of devices are sometimes referred to as "generic" SCSI devices. The generic SCSI BIOS and ABIOS provide an adapter-hardware-independent software interface to allow device drivers running under DOS and OS/2 to support applications using SCSI devices other than fixed disks.

The SCSI adapter BIOS interfaces are described in detail in the *SCSI Supplement to the IBM Personal System/2 and Personal Computer BIOS Interface Technical Reference*.

SCSI Peripheral Devices

Many SCSI peripheral devices are available today from dozens of different manufacturers. It is likely that there will be many more new SCSI devices developed in the coming years. Because of the device-independent architecture of the SCSI standard, SCSI adapters will be able

to support devices not even conceived of today and take advantage of new technology developments in current SCSI device types without requiring new adapters and additional Micro Channel adapter slots. Although most SCSI devices of the same type (for example, tape drives and disk drives) use compatible commands, some devices may use different or vendor-unique commands, or they may interpret commands differently from the way another device of the same type interprets the commands.

With the SCSI standard's flexibility comes the possibility of incompatibility with some devices that were developed before the official SCSI standard was developed, or they

*The flexibility of
the SCSI subsystem
allows for a
wide choice of
devices and
related applications.*

were developed with a different interpretation of some aspect of the SCSI standard, or they have not been tested in all possible configurations. Therefore, when choosing SCSI devices, take care to choose devices that have been previously tested with the adapter and software you are using, or reserve the right to return the device if it fails to operate properly in your configuration.

Different types of SCSI devices usually require different amounts of operating-system support software

and/or device drivers supplied either by the operating system, application program, or SCSI-device manufacturer. As mentioned in the SCSI BIOS section of this article, SCSI fixed disk operation under DOS and OS/2 is supported by the SCSI adapter's BIOS without the requirement for additional support software. IBM intends to provide the support software necessary for IBM SCSI devices other than fixed disks with the individual SCSI device options as they become available from IBM.

Third-party device and software vendors are encouraged to develop products that are compatible with IBM SCSI adapters. The *IBM Personal System/2 Micro Channel SCSI Adapter Technical Reference*, the *IBM Personal System/2 Micro Channel SCSI Adapter with Cache Technical Reference* and the *SCSI Supplement to the IBM Personal System/2 and Personal Computer BIOS Interface Technical Reference* manuals provide the necessary information for third-party SCSI device and SCSI support software developers to create products that are fully supported by the IBM SCSI adapters.

SCSI Subsystem Hardware and Software Configuration

The flexibility of the SCSI subsystem allows for a wide choice of devices and related applications. A SCSI adapter can either come standard in a PS/2 or can be added as an option adapter to other PS/2 Micro Channel systems. Up to four SCSI adapters can be installed in a single system.

The primary differences between the two IBM SCSI adapters are performance and cost. If the application programs being run on a

Hardware:	Software:
• PS/2 Micro Channel system	• IBM DOS 3.30 or above
• SCSI Adapter	• CD-ROM DOS device driver *
• SCSI internal cable	• Installable file system for CD-ROM "High Sierra" file format support *
• SCSI fixed disk	• CD-ROM based and other applications
• SCSI external cable	
• SCSI CD-ROM drive with external expansion chassis	
* The CD-ROM device driver and file system software are normally supplied with the CD-ROM drive.	

Figure 4. Requirements for a SCSI-Based System with a CD-ROM Drive

system use the SCSI fixed disk intensively, then the 32-bit cached adapter will usually provide improved fixed-disk performance because of its transparent, intelligent disk-caching feature. Also, with its 32-bit Micro Channel data path, the 32-bit cached adapter can also accomplish the data transfers to and from system memory in half the time required by the 16-bit adapter. This results in improved Micro Channel bus utilization and more efficient system operation. However, since the overall command processing time for most devices is much longer than the data transfer time on the Micro Channel, the use of a 32-bit data path does not generally result in a doubling of subsystem performance.

A high-performance, SCSI fixed disk should also be used if improved fixed-disk performance is desired. The 16-bit adapter can be used equally well in systems where fixed-disk performance and Micro Channel bus utilization are not critical to overall system performance.

For attaching most other SCSI devices, both adapters work equally well.

Let's look again at the PS/2 system shown in Figure 2. This system has a relatively simple subsystem configuration. Let's assume that the owner of this system wants to use the CD-ROM drive for running standard CD-ROM-based application programs, such as dictionary, thesaurus, desktop publishing graphics, and recreational titles under the DOS operating system. The SCSI fixed disk will be used to store the operating system, various application programs and other data. Let's also assume the 16-bit SCSI adapter has been chosen for this particular application because the proposed applications do not require intensive use of the fixed disk. The hardware and software requirements needed for this system are listed in Figure 4.

To allow up to eight SCSI devices to share a common SCSI adapter, each device must be assigned a unique identification (ID) number,

usually called a SCSI ID or SCSI address. For the IBM SCSI adapters, the adapter ID is set by the Programmable Option Select feature of the PS/2 computer. The default value for the adapter ID is 7. This leaves the numbers 0 through 6 for assignment to SCSI peripheral devices. The IBM SCSI subsystem convention is that the SCSI fixed-disk devices are assigned IDs in descending order starting with 6. These devices may be assigned IDs in any sequence, independently of the order in which they are connected to the bus, although no more than one device should be assigned the same ID. If two or more SCSI devices need to use the bus at the same time (for example, under a multitasking operating system such as OS/2), the device with the highest ID number will be allowed to use the bus, and the other devices will have to wait until the device with the highest ID number is finished using the bus.

For our example system, the SCSI adapter will be assigned ID 7, the fixed disk will be assigned ID 6, and the CD-ROM drive will be arbitrarily assigned ID 3. The fixed disk ID is normally set on the device by switches or jumpers. These are usually preset at the factory to ID 6. The external SCSI CD-ROM drive normally has an ID assignment switch located at the rear of the unit. Its ID switch should be set at position 3.

You can use the Configure SCSI Devices option of the PS/2 Reference Diskette to view the present SCSI subsystem configuration if you are unsure of the subsystem configuration or ID assignments. The Configure SCSI Devices option displays a list of the installed SCSI adapters and the attached SCSI devices and indicates the adapter and device

type, IDs, and storage capacity of each fixed disk.

Before installing the SCSI adapter in a Micro Channel adapter slot in the PS/2 system unit, the termination resistor chip located in a socket on the 16-bit adapter must be removed with a small screwdriver or an integrated-circuit-chip remover. This is necessary because this system configuration (that is, an internal fixed disk and an external CD-ROM drive) requires both internal and external cables. For proper operation, the cable should have termination resistor circuits (or terminator plugs) located at each end of the cable or within the devices located at each end of the cable.

If both connectors on the 16-bit adapter are not used in a configuration, then the termination resistor chip is placed in the socket on the 16-bit adapter. If the 32-bit adapter is used, any unused cable connector on the adapter must have an appropriate terminator plug attached. See the 32-bit SCSI adapter installation instructions for more information.

After installing the SCSI adapter, the internal SCSI fixed disk should be set to ID 6, and SCSI bus termination resistors should be installed on the fixed disk, or a termination plug should be installed on the end of the internal cable. The internal cable should then be connected between the adapter's internal connector located on the top edge of the card and the fixed disk's connector. Proper orientation of the bus cable (for example, connector pin 1 on the adapter is connected to pin 1 on the device) is essential for proper subsystem operation. Most cable connectors are keyed for proper orientation and easy installation. Next, the external CD-ROM drive should be set to ID 3 and connected to the

adapter by the external cable. A termination plug should be installed on the unused connector on the CD-ROM drive. Most external SCSI devices have two connectors to allow "daisy-chaining" of additional external SCSI devices with additional cables. The last device in a chain should have a terminator plug installed on the unused connector.

Once the system hardware has been properly installed, the PS/2 Reference Diskette is then used to complete the installation process by setting the adapter's ID, enabling the BIOS on the SCSI adapter, and determining the subsystem configuration, which enables automatic checking of the subsystem after each system reset and provides for system diagnostic support.

*Most external
SCSI devices allow
"daisy-chaining"
of additional
external SCSI devices.*

After system hardware installation and configuration is complete, DOS can be installed on the fixed disk according to the standard DOS installation procedure for fixed disks. The BIOS on the adapter makes the fixed disk appear to DOS the same as any other fixed disk. The CD-ROM device driver must be copied to the fixed disk and a "device=" statement added to the DOS CONFIG.SYS file to cause DOS to load the CD-ROM device driver, which enables DOS-level access to the CD-ROM drive. The CD-ROM install-

able file system should also be installed on the fixed disk according to the installation instructions provided with the CD-ROM support software. Because the way data is organized on a compact disk is different from the way DOS normally reads data from and writes data to a disk, the CD-ROM installable file system is required to allow DOS to manage the CD-ROM data files.

SCSI Subsystem Operation

For the example system in Figure 2, operation of the fixed disk is accomplished by command requests to the SCSI adapter. The commands sent to the fixed disk are generated by the BIOS on the SCSI adapter in response to requests from DOS and application programs. Operation of the CD-ROM drive is similar; however, the CD-ROM command requests to the adapter are generated by the generic BIOS on the SCSI adapter in response to requests from the CD-ROM device driver. The CD-ROM device driver receives requests from the CD-ROM installable file system, which gets its requests from the CD-ROM application program or DOS. This software request structure may be different for different operating systems and applications.

IBM SCSI adapters and attached devices are controlled primarily by a protocol consisting of control blocks stored in system memory by the application program, operating system, device-driver software, or by the SCSI adapter BIOS and status blocks stored in system memory by the SCSI adapters.

In general, when a SCSI device operation is desired, the appropriate software builds a Subsystem Control Block (SCB) in memory. This SCB contains the details of the com-

mand (for example, read five sectors of data from a fixed disk starting with sector number 591). The software then loads the adapter command interface registers with the 32-bit address of the SCB's location in system memory and initiates the command by loading the adapter's Attention register. Note that only the BIOS on the SCSI adapter should perform operations directly with the adapter hardware and registers. Higher-level software should use the BIOS interfaces to send command requests to the adapter.

Using its Micro Channel bus master capability, the SCSI adapter reads the SCB and performs the requested operation. After the operation is completed by the adapter, a Termination Status Block (TSB) is stored in system memory by the adapter to indicate the results of the operation, and a hardware-interrupt request is sent to the system processor to notify the software that the command has been completed, and the TSB has been stored. For improved performance, a TSB is usually stored only if the adapter detects an error while performing an SCB command.

The SCSI adapter SCB and TSB formats and other related programming information can be found in the *IBM Personal System/2 Micro Channel SCSI Adapter Technical Reference*, the *IBM Personal System/2*

Micro Channel SCSI Adapter with Cache Technical Reference and the *SCSI Supplement to the IBM Personal System/2 and Personal Computer BIOS Interface Technical Reference*. A general discussion of Subsystem Control Blocks can be found in the article on SCB in this issue.

Summary

The IBM PS/2 Micro Channel SCSI Adapters provide a general-purpose interface for attaching industry-standard SCSI devices to PS/2 Micro Channel-based systems. The adapters' Micro Channel bus master capability allows efficient use of the SCSI adapter and effective overlap of up to 15 SCSI logical units without assistance from the PS/2 system processor. With only one adapter and Micro Channel adapter slot required for overlapped operation of up to 15 SCSI devices, these adapters are not only cost-effective, but they also allow efficient use of the system's Micro Channel adapter slots.

Note: A copy of the SCSI-1 standard, "Small Computer System Interface X3.131-1986," is available from:

American National Standards
Institute
1430 Broadway
New York, NY 10018
(212) 642-4900

ABOUT THE AUTHOR

Andrew B. McNeill is a senior engineer in the Storage Controller Development area for the Entry Systems Division of IBM. He is currently designing SCSI subsystems and has previously designed fixed-disk subsystems for the IBM Series/1 minicomputer system. He holds several patents and has publications in numerous issues of the "IBM Technical Disclosure Bulletin" relating to storage subsystem design, disk caching, and imbedded controller-based design. Andy joined IBM in Boca Raton in 1979 after receiving a bachelor of science degree in electrical engineering from Clemson University in Clemson, South Carolina. He is a member of the Eta Kappa Nu and Tau Beta Pi engineering honor societies.

PS/2 Wizard Adapter

*Tina M. DeAngelis
IBM Corporation
Boca Raton, Florida*

PS/2 Wizard is a bus master card that incorporates the Intel i860 chip. In combination with Micro Channel architecture, it is a high-speed attached processor for numeric-intensive applications.

One of the most exciting new technologies announced this year will soon be available as a bus master for PS/2 Micro Channel computers. When Intel's i860 microprocessor was announced in February 1989, its high-performance capabilities were demonstrated on a prototype adapter in a PS/2 Model 80. This adapter, PS/2 Wizard, provides an application accelerator for numeric-intensive applications.

The design of the Micro Channel computers paved the way for this PS/2 numeric-intensive solution to be implemented quickly and efficiently. The combination of Micro Channel architecture and the Intel i860 microprocessor expands the PS/2 opportunity in new and existing areas. The benefits derived from this combination translate into performance improvements for existing PS/2-based applications and a new platform for technical workstation applications.

The PS/2 Wizard adapter, jointly developed by IBM and Intel, will be marketed exclusively by IBM. When Intel first approached IBM with the i860, its value as a high-performance numeric processor was immediately recognized. What remained was deciding how to pro-

vide the i860 microprocessor in a PS/2 computer. After weighing the factors, the decision was made to bring the i860 to the PS/2 as an attached processor that would take advantage of Micro Channel architecture. This solution required minimal operating system changes, and allowed focus on those applications that would benefit most from the performance of the i860 while never losing compatibility with the broad scope of PS/2 applications available today.

Once the implementation mechanism was determined, development efforts went into full swing immediately. Intel designed and developed the adapter, while IBM focused on the functions required to provide the bus master capabilities. This coordination of efforts by groups from

both companies forged the path for the i860 Micro Channel bus master.

The adapter (Figure 1) consists of three IBM-designed bus master interface chips, Intel's i860 microprocessor, 2 MB of memory, and support logic. The i860 microprocessor is the key component of the PS/2 Wizard adapter. With its one million transistors, this single VLSI component specializes in integer, floating-point and graphics calculations. The chip also has an internal 8 KB data and 4 KB instruction cache. Often referred to as a "super-computer on a chip," it performs approximately twice as fast as other reduced instruction set computer (RISC) chips running the Dhrystone benchmark (see Note).



The bus master chips give the card the capability of moving data into and out of system memory and accessing the entire system memory map and I/O address space. The private memory on the card provides 85-nanosecond access with 45-nanosecond static column feature. Two megabytes reside on the card, and a top card connector allows a daughter card to be attached with six additional megabytes. The powerful capabilities offered by the i860 microprocessor, together with the bus master capabilities of the adapter itself, merge into the attached processor that is an application accelerator for PS/2 computers.

An application accelerator, while much like a coprocessor, lets the application decide what to process where. In the case of the PS/2 Wizard adapter, sections of application code requiring extensive floating-point or integer operations or needing graphics assists will be directed to the i860 for execution. I/O requests should be limited in these code segments in order to allow the i860 to perform at its best.

The operating system support (Figure 2) required to handle this structure is implemented as a device driver. At the application's request, secondary pieces of the application that are to be executed by the i860 are moved across the Micro Channel by the device driver running on the 386™ or 486™ to the device driver running on the i860. A small kernel of code on the i860 then executes the code segment and returns the results through the device driver mechanism across the Micro Channel bus to the primary application on the 386 or 486. These secondary segments of code may be a small or large portion of the application, but execution must be initiated on the 386 or 486 processor. In addition, the 386 or 486 processor can be executing other applications. In OS/2, the i860 application can be just another application running in a Presentation Manager window. If real-time graphics updates are required, it is best to use a dual-screen configuration and use the bus master capabilities of the PS/2 Wizard adapter to write directly to the display adapter (Figure 3).

As an application accelerator, the PS/2 Wizard adapter fits best into what is commonly called the numeric-intensive computing (NIC) application area. Typical applications include molecular modeling, bond trading, seismic analysis, CAD/CAM, and medical imaging. These types of applications generally have a core section of code that handles the numeric manipulation and will show significant performance gains when processed by a high-speed processor like the i860. Application porting, in this case, requires a small amount of rework and a recompilation in order to take advantage of the i860. The two major criteria in selecting application candidates to port to the PS/2 Wizard adapter are those that have identifiable segments of numeric-intensive code and do not require a significant amount of I/O within the segment. When the applications have been recompiled, they will execute with PS/2 Wizard in PS/2 32-bit Micro Channel machines.

Micro Channel architecture, in general, is much more suited to the capabilities of the PS/2 Wizard

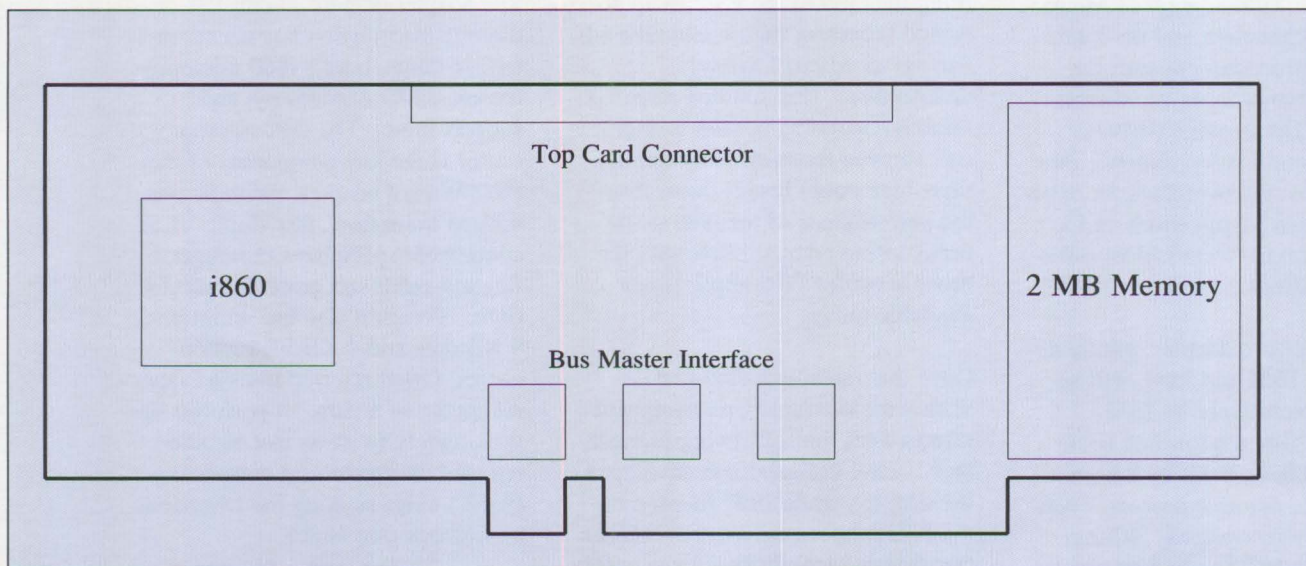


Figure 1. PS/2 Wizard Adapter

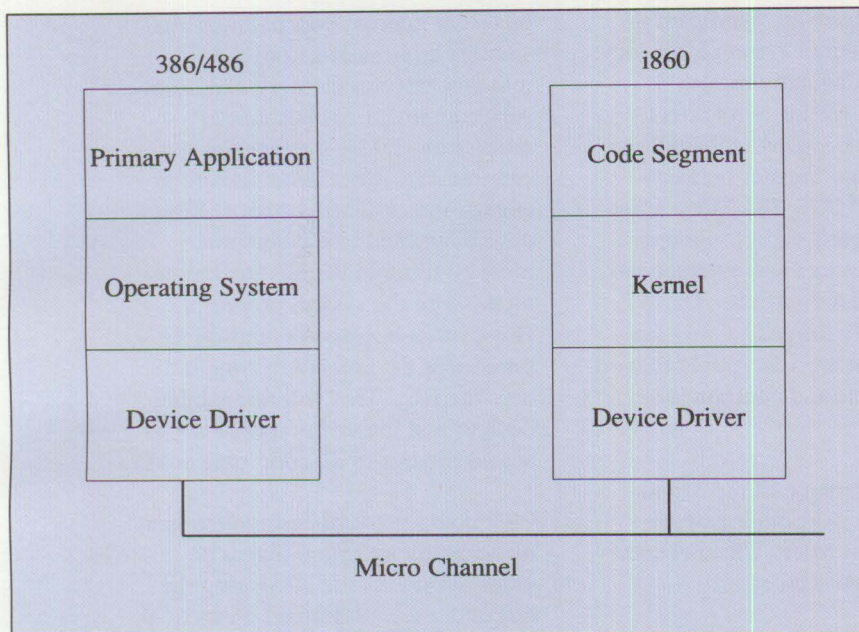


Figure 2. Operating System Support

adapter than is the AT bus structure. More specifically, two key factors brought about the decision to design PS/2 Wizard as a Micro Channel adapter:

- The 32-bit Micro Channel bus provides double the data transfer rate in the same cycle time as the AT bus and is, therefore, a much more efficient mechanism for passing the i860 the large amounts of data it is capable of processing.
- The ability of multiple bus masters to share the bus efficiently (that is, Wizard and a display adapter) increases the overall performance of the application. A key example is the performance improvements seen in applications requiring real-time updates to the screen.

The PS/2 Wizard adapter uses the power of Micro Channel architecture to bring the i860 microproces-

sor to the PS/2 line of products. In doing so, it extends the capability of personal computers into the workstation environment while continuing

to maintain the platform for existing personal computer applications.

Note: Additional information about the i860 microprocessor and the performance of the chip may be requested from Intel Corporation.

ABOUT THE AUTHOR

Tina DeAngelis joined IBM in 1979 as a programmer working with the EDX and RPS operating systems for Series/1. Next she moved into the PC software area of the business, working in PC Productivity Applications. She became manager of an operating system development group responsible for a vendored graphics software package, then she worked on a team that provided the OS/2 and PS/2 technical seminars for application developers. Tina is currently an advisory planner with operating system and application responsibility for the PS/2 Wizard adapter.

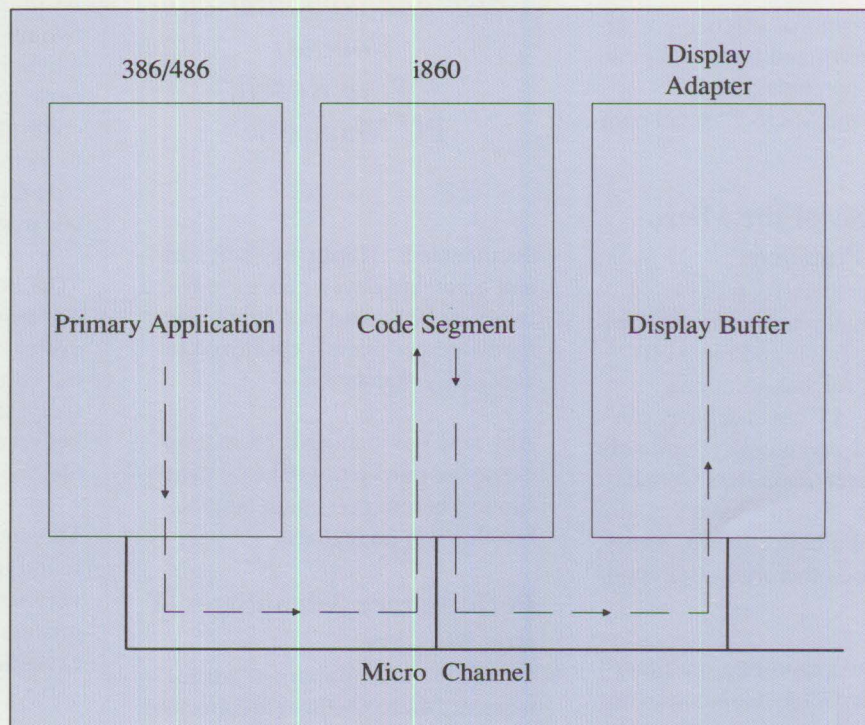


Figure 3. Writing Directly to the Display Adapter

Experience in Bus Master Design

*CORE International
Boca Raton, Florida*

CORE International has developed bus master adapters for PS/2 Micro Channel computers. Enthused by that experience, they offered to share their thoughts on Micro Channel architecture. This article was contributed by CORE.

With the PS/2, IBM delivers a microcomputer that goes beyond the AT bus standard of microcomputing. Technology breakthroughs of the Micro Channel architecture include optimum compatibility with OS/2 multitasking, full 32-bit processing, improved direct memory access, and balanced system performance.

CORE International offers high-performance drives and bus master controllers that maximize the I/O potential of the Micro Channel computers.

Advantages of the Micro Channel Platform

From our perspective, today's Micro Channel platform implements multitasking faster and more efficiently than AT bus solutions.

While many AT bus machines can run the OS/2 operating system, only Micro Channel computers were designed for it. Micro Channel is also the preferred architecture for Xenix, NetWare, and other multiuser applications.

In addition to supporting multitasking, the Micro Channel bus moves twice as much data as a 16-bit data

path. This leads to significant increases in overall system efficiency. The Micro Channel bus also eclipses the AT bus with direct memory access (DMA) that allows for high-speed transfer between memory and I/O. The Micro Channel bus supports eight programmable DMA devices and automatically resolves DMA conflicts. The AT bus is usually limited to only four channels that are susceptible to hardware conflicts and configuration problems.

Another advantage of the Micro Channel bus over other new advances is that Micro Channel specifications have been clearly

Potentially, the effective data throughput between disk and computer can double.

*—Winn Rosch,
PC Magazine**

documented. Third-party software and hardware developers have had a two-year head start designing new hardware peripherals optimized to exact PS/2 standards.

Any new bus standard can require extensive third-party development support before maximum tangible benefits can be realized.

Performance Advantages of Bus Masters

One of the most important advantages of Micro Channel architecture is its bus master capability. The

bus is an internal data pathway that shuttles information from the system processor to memory and accessory equipment. A bus master adapter has DMA intelligence and can transfer data without having to engage the system processor. Every device attached to the computer must go through the bus to communicate with the system processor. However, a bus master controller minimizes the amount of time it uses the bus. This will allow other devices and the system processor to access the bus to do additional work.

PS/2 computers are designed to handle as many as 15 intelligent devices or processors, all sharing the bus under a well-defined protocol allowing for equal access to the bus. This contrasts with the AT bus where fewer devices are supported and arbitration for control of the bus is not well-defined, sometimes resulting in unequal use of the bus.

As illustrated in Figure 1, it takes two microprocessor cycles to move a byte in a conventional bus — one cycle to move it to the DMA-slave controller, then a second to move it to memory. A bus master moves a byte directly to memory — in only one system-processor cycle!

"The bus master can make the memory move in half the time (one bus cycle)," said Winn Rosch, contributing editor of *PC Magazine*. "Potentially, the effective data throughput between disk and computer can double."

This increased performance, not found in AT bus systems, can be very significant because the system processor must handle millions of executions per second.

Balanced System Performance

IBM delivers a cost-effective, general-purpose computer that appeals to a broad range of end users. People buy high-speed 386 and 486 machines for high-performance solutions. But for maximum performance in most commercial multi-user systems that have to move a lot of data, users need optimized I/O for balanced system performance.

That's why CORE developed high-performance disk/bus master controller subsystems for disk-intensive applications. The 15 MHz bus master controller/ESDI disk subsystem, designed specifically for Micro Channel computers, delivers transfer rates of up to 1,500 KB per second.

According to Rosch, "as the first bus master disk subsystem for the

Micro Channel architecture, [CORE's solution] ushers in an entirely new technology. You may be

You may be tempted to buy this pairing for your PS/2 model 60 or 80 just because it's among the fastest mass-storage combinations around.

*—Winn Rosch, PC Magazine**

tempted to buy this pairing for your PS/2 Model 60 or 80 just because it's among the fastest mass-storage combinations around."

Networking

Duplexed drives in a server lessen the risk of having data lost because of a defective hard disk. IBM's controllers do not allow the duplexing of hard disks and multiple IBM controllers cannot co-exist in the same system. CORE's bus master hard disk controller provides the duplexing capability, as well as making PS/2 computers the highest performance solution. Maximized throughput with 15 MHz transfer rates as well as improved overall system performance inherent in bus mastering can be realized.

See Figure 2 for a listing of compatible peripherals.

* Reprinted from *PC Magazine* April 11, 1989.

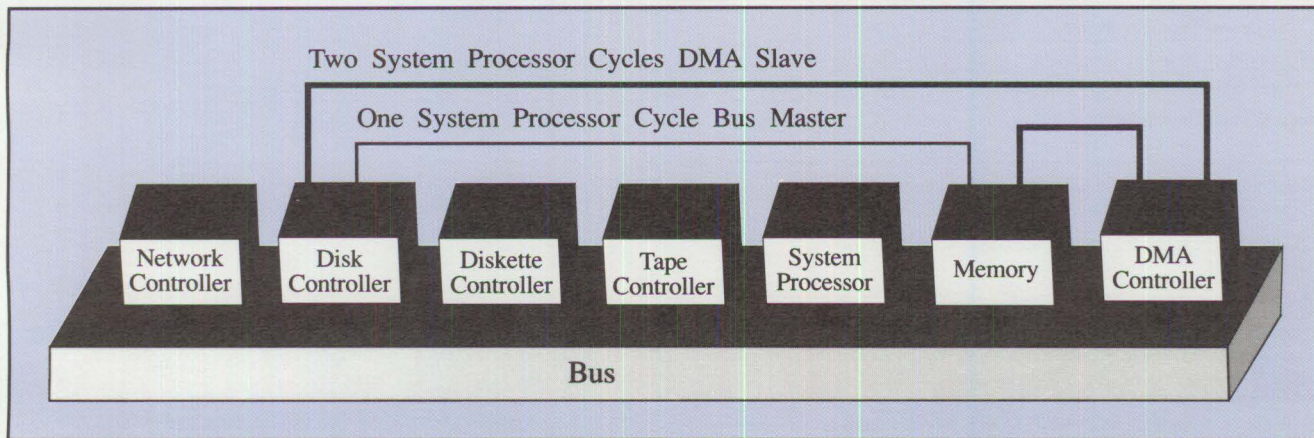


Figure 1. DMA and Bus Master Comparison

CORE Upgrade	25/30	30/286	50*	50Z	60	70	80
Hard Disk Drives							
OPTIMA 44 MB ST506 3.5", 21 ms, 5 MHz	E	I/E	I/E	N/A	I/E	N/A	I/E
OPTIMA 80 mb ST506 3.5", 15 ms, 5 MHz	E	I/E	I/E	N/A	I/E	N/A	I/E
HC Series 175 MB ESDI 3.5", 15 ms, 10 MHz	N/A	I/E	I/E	E	I/E	E	I/E
OPTIMA 40 MB ST506 5.25", 26 ms, 5 MHz	E	E	E	N/A	I/E	N/A	I/E
OPTIMA 70 MB ST506 5.25", 26 ms, 5 MHz	N/A	E	E	N/A	I/E	N/A	I/E
HC Series 90 MB ESDI 5.25", 16 ms, 10 MHz	N/A	E	E	E	I/E	E	I/E
HC Series 100 MB ESDI 5.25", 9 ms, 10 MHz	N/A	E	E	E	I/E	E	I/E
HC Series 150 MB ESDI 5.25", 16 ms, 10 MHz	N/A	E	E	E	I/E	E	I/E
HC Series 310 MB ESDI 5.25", 16 ms, 10 MHz	N/A	E	E	E	I/E	E	I/E
HC Series 380 MB ESDI 5.25", 16 ms, 10 MHz	N/A	E	E	E	I/E	E	I/E
HC Series 650 MB ESDI 5.25", 16 ms, 15 MHz	N/A	E	E	E	I/E	E	I/E
Backup Tape Systems							
CT 40 MB, transfer rate: 50 KB/sec	N/A	I/E	I	N/A	I	N/A	I
CT 60/100 MB, transfer rate: 90 KB/sec	E	E	E	E	I/E	E	I/E
CT 150/250 MB, transfer rate: 112.5 KB/sec	E	E	E	E	I/E	E	I/E
CT 500 MB Dual, transfer rate: 112.5 KB/sec	E	E	E	E	I/E	E	I/E
CT GIGAfile 2.2GB, transfer rate: 246 KB/sec	N/A	E	E	E	I/E	E	I/E
CT 125 MB, transfer rate: 100 KB/sec	E	E	E	E	E	E	E
Disk and Tape Controllers							
CNT-MCA Bus Master Controller, 15 MHz	N/A	N/A	X	X	X	X	X
Micro Channel Tape Host Adapter	N/A	N/A	X	X	X	X	X

Key: N/A = not available; E = external; I = internal

* The PS/2 Model 50 has been withdrawn from marketing.

Products listed are registered trademarks of their respective manufacturers. CORE International, CORE, COREtape, GIGAfile, HC Series, and OPTIMA are registered trademarks of CORE International, Inc.

Printed in the USA. All rights reserved.

Copyright 1989 CORE International

Figure 2. CORE Peripherals Compatibility Chart

Bus Master Adapters from Independent Option Vendors

*Mili Sanderson
IBM Corporation
Boca Raton, Florida*

This is a listing of bus master adapters that have been developed by independent option vendors for IBM PS/2 Micro Channel computers.

At Spring / COMDEX® '89, eleven bus master adapters from independent option vendors (IOVs) were on display in the IBM booth. Five developers participated with IBM in demonstrations of their products. Today, over 93 IOVs have secured 197 Micro Channel bus master card IDs from IBM. These numbers indicate a growing interest in developing adapters for the Micro Channel. Twenty-seven bus masters for PS/2 computers have been enrolled with IBM. A wide product mix is represented:

10 Communications Adapters
2 Storage Product
6 I/O Adapters
1 Data Acquisition Product
3 Display Adapter
5 Other Products

This listing provides individual descriptions of these bus master adapters. It was supplied to IBM by authorized representatives of the named IOVs.

ABOUT THE AUTHOR

Mili Sanderson joined IBM in 1974. She has held a variety of marketing support positions in Office Products, Data Processing, National Accounts and Entry Systems Divisions. She became a member of the PS/2 development planning group in 1983, and is now is part of the special marketing team assisting with IOV development for the Micro Channel. She is the primary interface to the IOV community, offering participation in IBM's Micro Channel publicizing efforts to vendors.

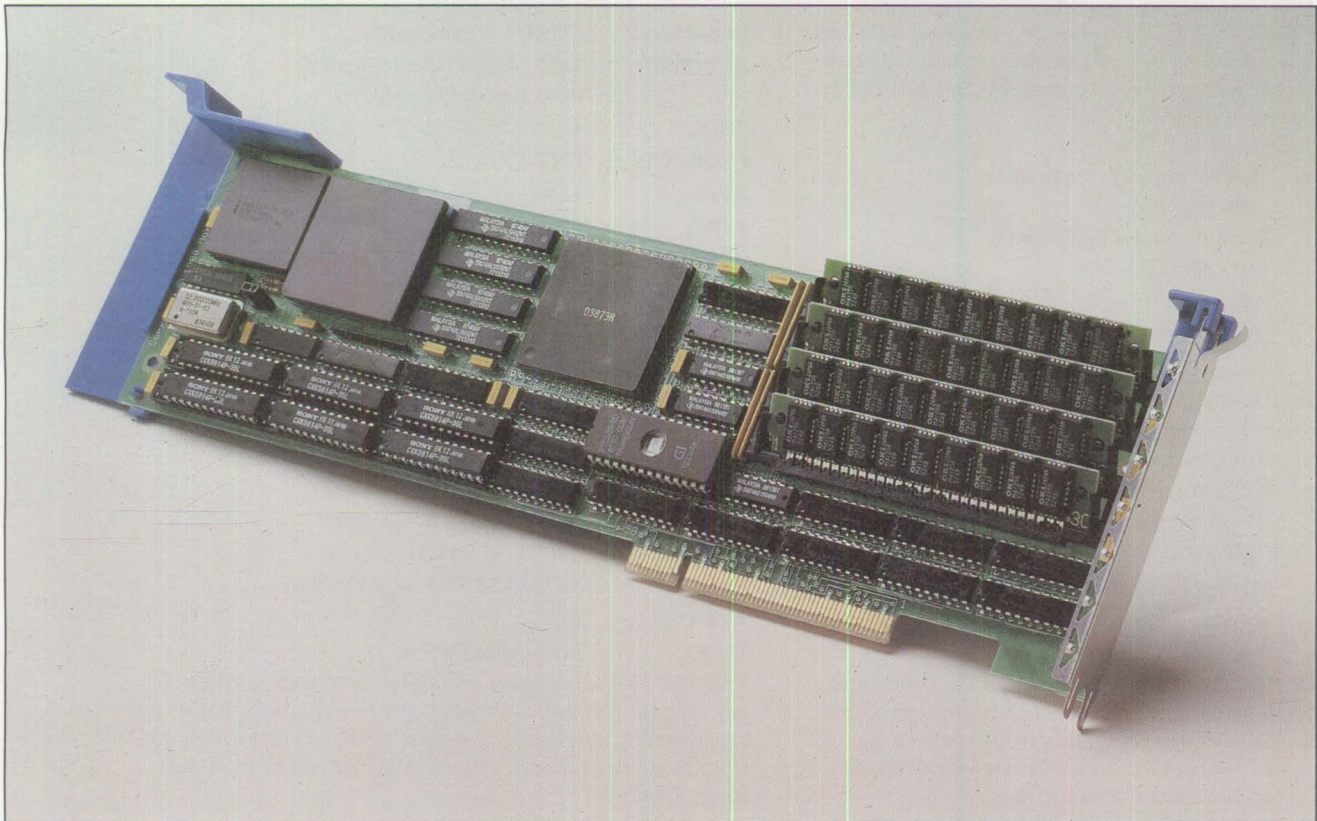


Figure 1. Aox Incorporated MicroMASTER 386 Card

COMPANY: Adaptec, Inc.
ADDRESS: 691 South Milpitas Boulevard
 Milpitas, CA 95035
PHONE: 408-945-8600
PRODUCT: ADAPTEC AHA-1640 SCSI HOST
 ADAPTER
AVAIL: 12/88
TYPE: SCSI I/O Attachment
DESCRIPTION: SCSI I/O channel supports CDROM, WORM, and magnetic devices. High performance multitasking mailbox architecture. Up to 120 active SCSI I/O processes with command queueing and command linking. Bus master data transfer up to 8 MB. 5 MB synchronous, and 2 MB asynchronous continuous SCSI data transfer. SCSI-2 external connector. Much more.

COMPANY: Aox Incorporated
ADDRESS: 486 Totten Pond Road
 Waltham, MA 02154
PHONE: 617-890-4402
PRODUCT: MicroMASTER™ 386
AVAIL: 7/89
TYPE: Memory Product – 32-Bit
DESCRIPTION: 80386-based processor upgrade bus master for IBM PS/2 Model 50, 50Z, and 60 computers (See Figure 1). Available in 20, 25, and 33 MHz versions. Supports 80387 numeric co-processor and up to 8 MB of 32-bit memory on card. Provides multiprocessing capability for all Micro Channel bus based PS/2 computer models.

COMPANY: Ariel Computer Corporation
ADDRESS: P.O. Box 866
 Flemington, NJ 08822-0866
PHONE: 201-788-9002
PRODUCT: Video One™
AVAIL: 11/89
TYPE: Image Capturing Product
DESCRIPTION: Image capture and rendition device.

COMPANY: BICC Data Networks, Inc.
ADDRESS: 1800 West Park Drive
 Westborough, MA 01581
PHONE: 800-447-6525
PRODUCT: 4110-3 ISOLAN ® Controller Card
AVAIL: 4/89
TYPE: Network Adapter
DESCRIPTION: A fully integrated high-speed PS/2 computer bus master controller providing Ethernet IEEE local area networking. Innovative design maximizes overall performance by utilizing the total capacity of Ethernet local area networking and IBM PS/2 computer bus master capability.

COMPANY: ComTech International, Inc.
ADDRESS: P.O. Box 722
 Bloomfield, CT 06002
PHONE: 203-286-8440
PRODUCT: Channel/2™
AVAIL: 9/89
TYPE: Host Communication Adapter
DESCRIPTION: An intelligent gateway for NetBIOS LANs with two channels. Supports up to 384 concurrent sessions over SDLC and/or X.25 communication line speeds of up to 64 Kbps on separate channels. LIM EMS 4.0 support for workstations.

COMPANY: ComTech Imaging Technology
ADDRESS: P.O. Box 722
 Bloomfield, CT 06002
PHONE: 203-286-8440
PRODUCT: Personal Image™
AVAIL: 3/90
TYPE: Image Capturing Product
DESCRIPTION: NetBIOS compatible LAN document storage, retrieval, and management. Picture/graphic, and freeze-frame video camera and scanning devices. LIM EMS 4.0 support at workstation. High speed TT links to mainframes.

COMPANY: CORE International®
ADDRESS: 7171 North Federal Highway
 Boca Raton, FL 33487
PHONE: 407-997-6055
PRODUCT: CNT-MCA™
AVAIL: 12/88
TYPE: Fixed Disk Drive
DESCRIPTION: 15 MHz ESDI hard disk controller, supports 2 drives. 2 controllers support duplexing under Novell® 2.15 or Netware® 386 software. Can coexist with IBM controllers. 1500 KB/second data transfer rate for 15 MHz drives or 10 MHz drives at 1100 KB/second.

COMPANY: ERACOM Pty Ltd
ADDRESS: 28 Greg Chappell Drive
 Burleigh Heads,
 Queensland 4220 Australia
PHONE: 011-61-75-934-911
PRODUCT: MC MASTER ENCRYPTOR
AVAIL: 11/89
TYPE: Encryption Device
DESCRIPTION: Intelligent encryption subsystem with secure key storage. Provides automatic disk encryption (optional) and programming interface; full set of utilities and user access control and authentication; COMMS daughter boards for Token-Ring, SNA, etc.

COMPANY: Graphic Software Systems, Inc. (GSS)®
ADDRESS: 9590 Southwest Gemini Drive
 Beaverton, OR 97005
PHONE: 503-641-2200
PRODUCT: GSS MC1000 Controller
AVAIL: 8/89
DESCRIPTION: The GSS MC1000 is the Micro Channel bus accelerator for graphics-based computing environments. Its custom hardware, bus master I/O, programmable graphics processor, and optimized firmware interfaces accelerate graphics user interfaces and applications in DOS, OS/2, and UNIX computer software.

COMPANY: HAAR Industries, Inc.
ADDRESS: 2600 Virginia Avenue, NW
 Washington, D.C. 20037
PHONE: 202-388-8550
PRODUCT: Smart Eight/2™
AVAIL: 6/88
TYPE: Serial Port
DESCRIPTION: Intelligent eight-port serial board with: C188 processor, 256 K dual ported RAM, clock, and boot. RJ11, RJ12, or RJ45. Multi-boards may be chained. Realtime OS & MS DOS compiler available to develop on-board applications.

COMPANY: I-BEAM, Inc.
ADDRESS: 303 Wyman Street
 Waltham, MA 02154
PHONE: 617-890-7080
PRODUCT: The I-BEAM Board
AVAIL: 11/89
TYPE: Display Adapter
DESCRIPTION: Anti-aliased character generation system, rendering screen more readable than hard copy, delivered on display adapter that drives VGA (640 x 480) and 1024 x 768 screens, bundled with font generation capability. Drivers available for Windows™ and AutoCAD® software and IBM PS/2 Color Display 8514.

COMPANY: IMC Networks Corp.
ADDRESS: 1342 Bell Avenue Suite 3E
 Tustin, CA 92680
PHONE: 714-259-1020
PRODUCT: PCnic MCAbus
AVAIL: 9/89
TYPE: Network Adapter
DESCRIPTION: 32-bit bus master Ethernet LAN card for use with PS/2 Model 70 and 80 computers. Works as a fileserver or workstation under Novell NetWare V2.1x and NetWare 386, has full POS registers for I/O address and IRQ level selection, adheres to IEEE 802.3 Ethernet standard plus supports 40, 75, and 93 Ohm coaxial cable.

COMPANY: Lantana Technology Inc.
ADDRESS: 4393 Viewridge Avenue, Suite A
 San Diego, CA 92123
PHONE: 619-565-6400
PRODUCT: Cypress/2
AVAIL: 2/89
TYPE: Network Adapter
DESCRIPTION: Token-Ring compatible local area network controller bus master implementation for high speed 802.2 and 802.5 compatible.

COMPANY: Madge Networks
ADDRESS: 1580 Oakland Road
 Suite C-206
 San Jose, CA 95131
PHONE: 800-876-2343
PRODUCT: MC Ringnode
AVAIL: 11/88
TYPE: Network Adapter
DESCRIPTION: High performance, bus master token ring network interface adapter including software support for PC LAN, LAN Server, LAN Manager, NetWare, and other NetBIOS-compatible LANs and installation diagnostics.

COMPANY: Madge Networks
ADDRESS: 1580 Oakland Road Suite C-206
 San Jose, CA 95131
PHONE: 800-876-2343
PRODUCT: SMART MC Ringnode
AVAIL: 11/88
TYPE: Network Adapter
DESCRIPTION: Bus master token ring network interface adapter with 128 KB RAM permitting LAN protocol software to run on the card freeing DOS memory. Software for PC LAN, LAN Server, LAN Manager, NetWare, and other NetBIOS-compatible LANs and installation diagnostics.

COMPANY: METACOMP, Inc.™
ADDRESS: 15175 Innovation Drive, Building A
 San Diego, CA 92128
PHONE: 619-673-0800
PRODUCT: PScomm2/4™
AVAIL: 2/89
TYPE: Serial Port
DESCRIPTION: High-performance bus master serial I/O controller provides two or four programmable channels. Built around an 80C186 16 MHz CPU; relieves the host processor of overhead associated with complex communications protocols. Supports most electrical / mechanical interface standards. OS/2 driver available.

COMPANY: METACOMP, Inc.
ADDRESS: 15175 Innovation Drive, Building A
 San Diego, CA 92128
PHONE: 619-673-0800
PRODUCT: PSconnect™
AVAIL: 10/89
TYPE: Serial Port
DESCRIPTION: Intelligent serial I/O communications subsystem supports async devices. One system card slot supports 127 eight-or-sixteen-port Remote Async Concentrators (RACs). Twisted pair RJ-45 connectivity. Hub controller with fiber optic ports. Bus master interface.

COMPANY: Mountain® Computer, Inc.
ADDRESS: 240 Hacienda Avenue
 Campbell, CA 95008
PHONE: 800-458-0300
PRODUCT: FileSafe™ Series 2100
AVAIL: 4/89
TYPE: Tape Drive
DESCRIPTION: An 8 mm helical scan tape drive with a maximum capacity of 2.2 GB. Gully compatible with IBM PC LAN and includes Mountain FileSafe software. Novell 2.15 certified.

COMPANY: Northern Telecom Inc.
ADDRESS: 2435 North Central Expressway
 Richardson, TX 75080
PHONE: 214-301-2000
PRODUCT: PCLink/MC
AVAIL: 3/89
TYPE: Network Adapter
DESCRIPTION: PCLink/MC allows PS/2 computer users to access Meridian DNS Network servers and services over a dedicated 2.56 Mbps twisted pair link.

COMPANY: Northern Telecom Inc.
ADDRESS: 2435 North Central Expressway
 Richardson, TX 75080
PHONE: 214-301-2000
PRODUCT: PCLink/MC64
AVAIL: 8/89
TYPE: Network Adapter
DESCRIPTION: PCLink/MC64 allows PS/2 computer users to access meridian DNS network servers and services over a dedicated 1.56 Mbps twisted pair link. Includes 64K buffer.

COMPANY: Pacific Image Communications, Inc.
ADDRESS: 111 South Arroyo Parkway, Suite 420
 Pasadena, CA 91105
PHONE: 818-441-0104
PRODUCT: SuperFax MC/IPS
AVAIL: 4/89
TYPE: Facsimile Product
DESCRIPTION: One of the first true bus master solutions running under Windows. Up to seven FAX boards can run concurrently in a true background mode with no degradation to the foreground task.

COMPANY: Pixelworks, Incorporated
ADDRESS: Seven Park Avenue
 Hudson, NH 03051
PHONE: 603-880-1322
PRODUCT: Ultra Clipper™ Graphics
AVAIL: 1/89
TYPE: Display Adapter
DESCRIPTION: High performance, 1024 x 768, 1280 x 1024 graphics co-processor. Controllers support the Micro Channel bus master, DMA slave operations and programmable interrupts, I/O ports, and DMA channels. The master function enables quick traversal of hierarchical display lists in PS/2 computer memory. Popular drivers include AutoCAD, VersaCAD, GEM™, etc. Full 2D, 3D integer and optional floating point transformations.

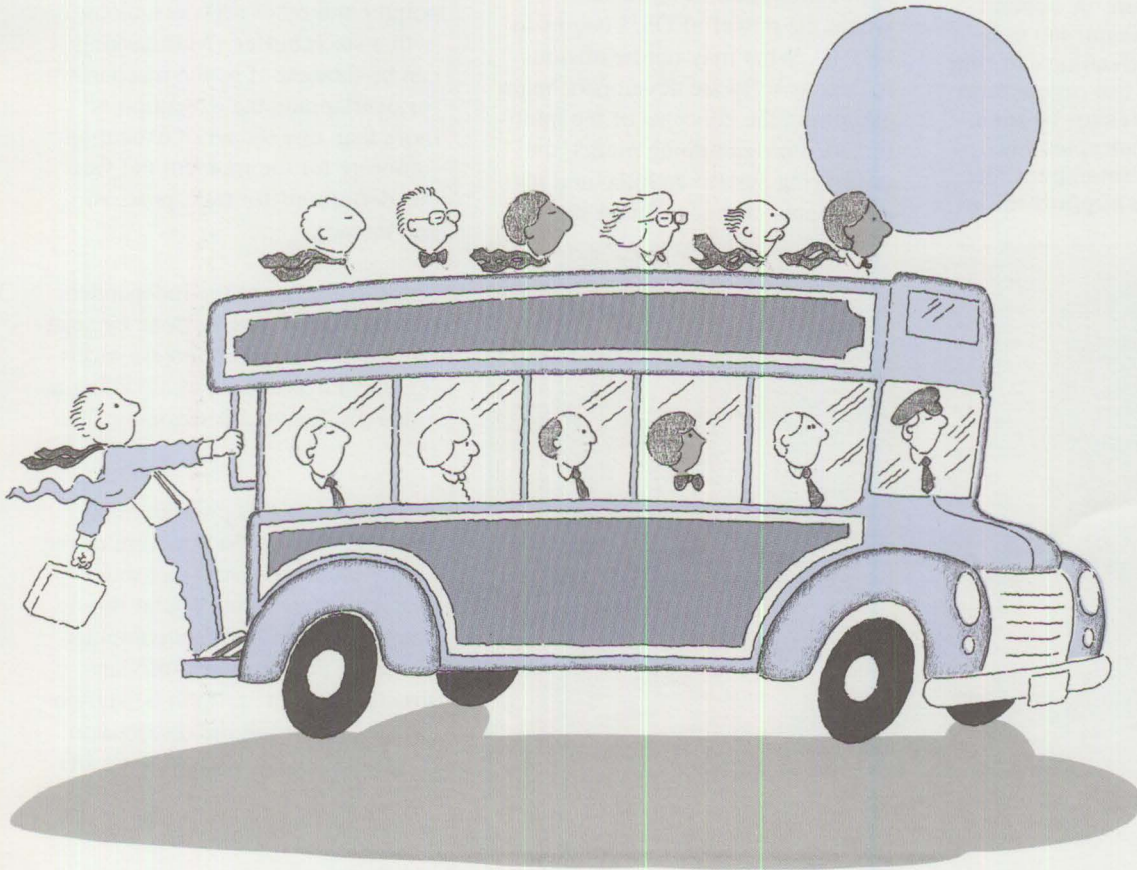
COMPANY: Proteon®, Inc.
ADDRESS: Two Technology Drive
 Westborough, MA 01581-5008
PHONE: 508-898-2800
PRODUCT: p1840 ProNET®-4 Interface
AVAIL: 1/89
TYPE: Network Adapter
DESCRIPTION: Connects PS/2 computers with Micro Channel bus to high performance, 4 MB/second ProNET®-4 Token-Ring network. Compatible with IEEE 802.5 specification and IBM Token-Ring. Uses 16-bit Micro Channel bus data path and POS feature for high speed and versatility. Compatible with Novell NetWare and IBM networking software.

COMPANY: Systems Solutions
ADDRESS: 12 Pinewood Court
 Newconset, NY 11767-2056
PHONE: 516-360-8879
PRODUCT: HA-1000
AVAIL: 9/89
TYPE: SCSI I/O Attachment
DESCRIPTION: High performance SCSI host adapter for 16/32-bit Micro Channel systems. Operates as bus master to achieve 5 MB per second SCSI data rates. Complete with driver and configuration software.

COMPANY: Texas Instruments
ADDRESS: P.O. Box 1443
 Mail Stop 706
 Houston, TX 77251-1443
PHONE: 713-274-3480
PRODUCT: TMDS380MCA
AVAIL: 1/89
TYPE: Network Adapter
DESCRIPTION: 16 MB per second Micro Channel bus master Token-Ring adapter. Compatible with IBM Token-Ring and IEEE 802.5 and IEEE 802.2.

COMPANY: Western Digital Corporation
ADDRESS: 2445 McCabe Way
 Irvine, CA 92714
PHONE: 714-863-0102
PRODUCT: WD7000-MSC
AVAIL: 5/89
TYPE: SCSI I/O Attachment
DESCRIPTION: Plug-in adapter board provides a fully functional interface between the Micro Channel bus of a PS/2 Model 50, 60, or 80 computer or compatible and a small computer systems interface (SCSI) bus.

COMPANY: YARC Systems Corporation
ADDRESS: 5655 Lindero Canyon Road Suite 721
 Westlake Village, CA 91362
PHONE: 818-889-4388
PRODUCT: Micro 785+
AVAIL: 8/89
TYPE: Co-processor
DESCRIPTION: The YARC Systems Micro 785+ is an advanced architecture co-processor featuring a Motorola MC68020. It operates at 40 MHz and offers five VAX MIPS of power to the professional user. The system supports FORTRAN, C, and Pascal compilers and applications, for example, MIRA from Oklahoma Seismic.



Bus Masters and OS/2

Bob Williams
IBM Corporation
Boca Raton, Florida

This article focuses on the bus master feature of Micro Channel architecture as it relates to the OS/2 multitasking operating system. The bus master feature allows full exploitation of the benefits of the overlapped I/O characteristic of a multitasking environment. These benefits are directly related to the concurrent programming paradigm (model). A multitasking application will generally isolate I/O logic to one or more separate I/O threads of processor execution. A system exploiting a bus master I/O subsystem can take advantage of this model and reduce the percentage of the system processor needed to drive the I/O. Examples discussed include an intelligent file I/O subsystem and support for an

intelligent graphics processor. The system configuration requirements and operating system services necessary to support such Micro Channel bus masters are also summarized.

New Opportunity for Overlapped I/O

The multitasking environment of the OS/2 operating system provides the user with many benefits, such as the ability to support the "messy desk" concept where multiple applications can be in various stages of execution. But executing multiple applications is just the beginning of the benefits of a true multitasking system. The ability to support reliable background execution of communications and data base applications and subsystems is where the power of OS/2 begins to be felt. What may not be obvious is that, as software developers begin exploiting the concepts of the multithreaded programming model, the underlying system architectural requirements change. This is one of

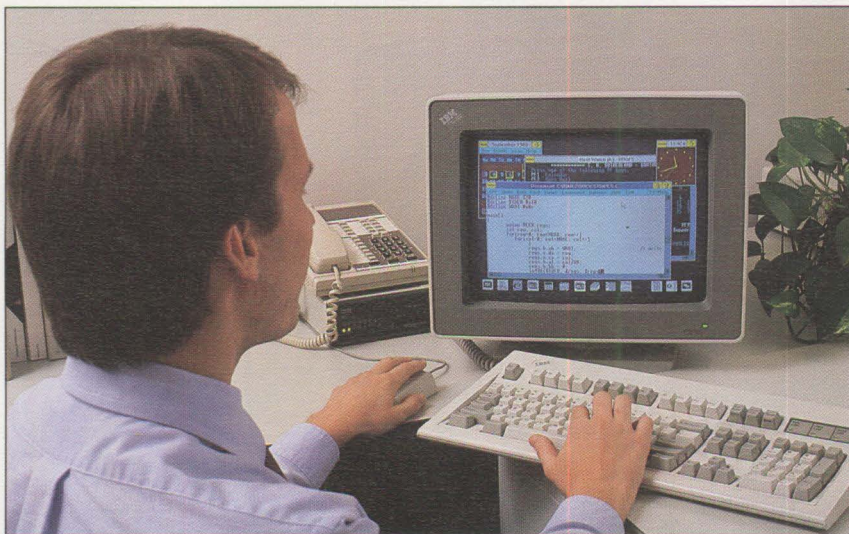
the areas in which Micro Channel architecture is a major breakthrough in the PC-compatible arena. The synergy between the bus master feature of Micro Channel architecture and the support structures and capabilities of the OS/2 multitasking programming environment is the subject of this article.

The Multithreaded Programming Model

The multithreaded programming model allows the isolated (asynchronous) processing requirements of software to conveniently execute independently using the concept of separate "threads of execution." A thread of execution can be thought of as a processor state – a snapshot of the values for the instruction pointer and other registers, along with a stack buffer. Multitasking can be thought of as the mechanism for overlapping the execution of more than one thread of execution (ignoring for the moment the various definitions for task, process, and thread).

Some examples of the independent processing that can be done by separate threads of execution are receiving user keyboard input, updating a video display, and performing file I/O.

In many ways this programming model simplifies the structure of the highly interactive software that is characteristic of the popular programs found in personal computer systems. Furthermore, software structured in this fashion is well-positioned to exploit the distribution of system processing. Moving I/O



execution responsibility out to intelligent I/O subsystems reduces the processing burden on the system processor. This is precisely the opportunity presented by the multiple bus master capability of Micro Channel architecture.

There are many publications available on both the fundamentals of concurrent programming and the specifics of the OS/2 multitasking environment. Instead of covering this same ground, this article looks beneath the application programming level. We use examples to highlight the system structures and services available to build intelligent I/O subsystems exploiting the bus master capability of Micro Channel architecture. To better understand the examples, let's look first at some important points about the system extension mechanisms and structures provided by OS/2.

System Structures - Dynamic Link Libraries and Device Drivers

OS/2 application programs generally take advantage of bus master devices through the use of a subsystem. Examples of two OS/2 subsystems are the file system and the Presentation Manager™ windowing and graphics services. These subsystems support extension mechanisms that allow customization of much of the device-specific support.

Primarily for performance reasons, the Presentation Manager is designed as a dynamic link library (dynamically bound subroutine) subsystem that executes within the

most protected and least privileged application privilege domain (Intel 80286 ring 3). Device-specific routines that need to do IN/OUT instructions to hardware registers can be designed to execute within the less protected I/O-privilege domain (Intel 80286 ring 2). A mechanism is needed to allow Presentation Manager to adapt to the unique programming interface of a specific graphics device. Presentation Manager is structured to support a replaceable device-specific module called a presentation driver. Such a presentation driver (also called a presentation device driver) is implemented as an installable dynamic link library module.

On the other hand, to preserve data integrity and for other reasons, the file subsystem is designed as a privileged extension of the operating system within the least protected and most privileged kernel-privilege domain (Intel 80286 ring 0). A customized set of routines allowing the file system to drive a specific device is contained in a structure called a device driver (also called kernel device driver or physical device driver). Device drivers are the components of the operating system that contain the hardware interrupt handlers.

Figure 1 shows the relationship of dynamic link libraries and device drivers to the rest of the system.

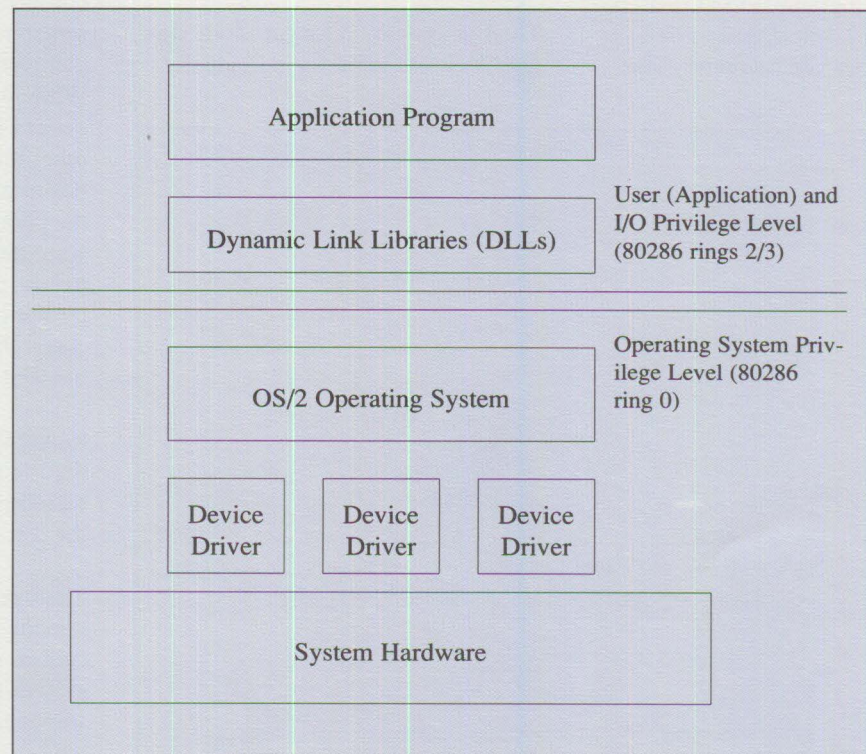


Figure 1. Relationship of Device Drivers and DLLs to System

With this brief description of two of the important OS/2 system extension mechanisms as background, let's look at some examples of intelligent I/O subsystems that exploit bus masters.

Intelligent I/O Subsystems

A DASD Device: One of the most obvious opportunities for exploiting the Micro Channel bus master capability is to support one of the most important subsystems in the operating environment – the file system. File I/O is often optimized within a file system using caching and asynchronous prefetch techniques. These techniques can help speed up individual file I/O operations from an application-program perspective. Unfortunately, they also tend to add additional system overhead to an already slow and expensive process. This overhead results regardless of whether direct memory access (DMA) or programmed I/O is used, because extra data is moved between memory and secondary stor-

age. The effect of this additional overhead can be minimized by using a bus master-based intelligent I/O subsystem to reduce the burden on the system processor.

In any computer system, the file system generally maps a user file I/O function call (read/write) into a series of block I/O requests to a direct access (secondary) storage device (DASD). On the Personal Computer AT class of machines, it was up to the disk/diskette device driver to translate these individual I/O requests into the low-level disk controller commands. Assume for this example that the file system is extended to support intelligent DASD subsystems such as those that are now practical with the bus master capability of the Micro Channel architecture. This intelligent DASD subsystem is supported by borrowing a concept common among mini and mainframe systems.

The file system no longer sends single-block I/O requests individually

to the disk device driver. Instead, it can build a list data structure that describes the entire series of block I/O requests corresponding to a user I/O function call (similar to System/370 channel programs). This data structure includes the appropriate read/write commands, pointers to user I/O buffers, and other relevant DASD management information. The device driver for the intelligent I/O adapter is then only responsible for initiating the I/O operation. This can be as simple as performing any necessary preparation of the user's buffers for DMA, and then passing the command data structure to the DASD bus master. At this point the device driver yields the system processor and allows the bus master adapter to run totally asynchronously to the rest of the system.

The thread in the device driver remains blocked (asleep) on some synchronization primitive (that is, semaphore) until the bus master completes the list of operations. While this I/O thread is blocked, the system processor is free to execute other threads in the system. The bus master signals completion of the command list by issuing a hardware interrupt. The device driver, having previously registered an interrupt handler, gets control and clears the semaphore to wake up the blocked I/O thread.

Figure 2 shows a representation of the file system and the DASD device driver and their relationship to the rest of the operating system.

The overall performance of multi-threaded software in this scenario increases as it receives additional system processor time for the non-I/O-bound threads. This is because the bus master enables the system to overlap the I/O with other opera-

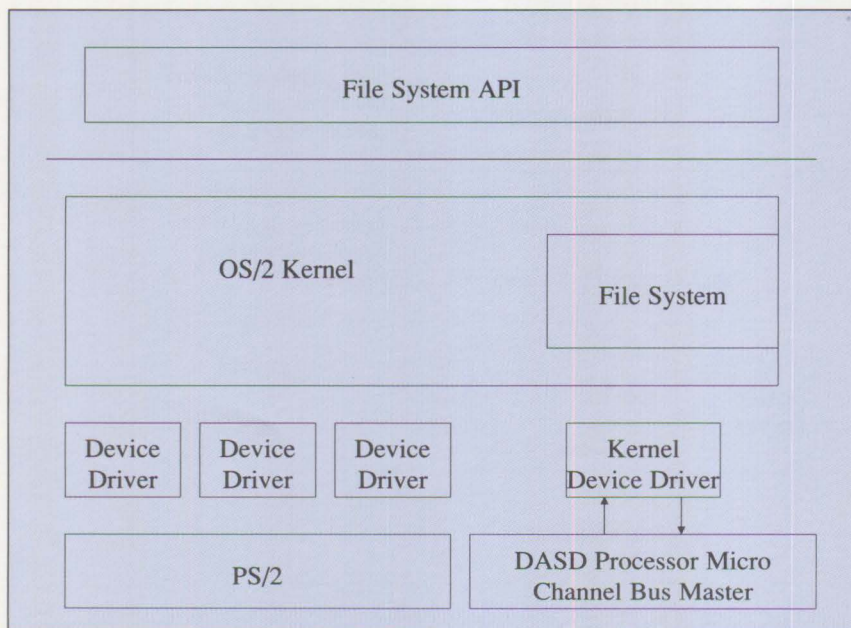


Figure 2. OS/2 Bus Master DASD Processor Subsystem

tions driven by the system processor. We will look in more detail at the services the device driver uses to manage a bus master adapter. But before looking at these services, let's briefly consider another type of intelligent I/O subsystem.

A Graphics Processing Device:

The same bus master concepts discussed in the previous DASD example can be applied to an intelligent graphics subsystem. Complex graphics operations can be system-processor-intensive, requiring manipulation and movement of large quantities of data. A thread within a graphics application (or text application using graphics fonts) can spend many system-processor time slices down in the video subsystem (that is, OS/2 Presentation Manager). Micro Channel architecture enables the development of an intelligent graphics I/O processor that can off-load much of the work from the system processor. Such graphics processors can be optimized for very fast raster- and/or vector-based graphics operations.

In the OS/2 Presentation Manager (PM) environment, a bus master subsystem more than likely includes at least two subcomponents. One PM presentation driver (video display or printer/plotter driver) component exploits the PM open architecture to register to receive control on certain graphics operations. These graphics operations can be offloaded to the bus master instead of executing the corresponding function in the system-processor-driven Presentation Manager graphics engine software.

The second component of the bus master subsystem is an OS/2 device driver that registers a hardware interrupt handler, as in the previous DASD example. This component is needed to receive asynchronous noti-

fication from the bus master of graphics operation completion. As longer and more complex drawing instructions are passed to the bus master video adapter, increased concurrency is achieved between graphics processing and other functions of the system. We should note that the designer of bus master graphics-processor subsystems will have to make important trade-offs of video throughput versus user-input responsiveness.

*Graphics operations
can be offloaded
to the bus master
instead of executing
in the system processor.*

Graphics operations that do not directly affect the console display are more easily optimized. For example, a single large and/or complex operation can be issued to the graphics processor to prepare an off-screen bit map for printing. Other considerations come into play in this case. One consideration is making sure that such an operation is interruptible and preemptible by a higher priority request to the graphics processor to update the console. Another consideration is balancing the resident memory requirements of a large off-screen bit map against the needs of the rest of the system.

Figure 3 shows a representation of the operating system structures discussed in this graphics processing example.

Intelligent I/O Subsystem Possibilities

There are many other examples of bus master subsystems that we could have used. The DASD subsystem could be generalized to support a Micro Channel bus master adapter that attaches a small computer system interface (SCSI) bus. The bus master feature is also well-suited for intelligent data communications adapters. Such an adapter could be designed to support multiple protocols. Similarly, it is now quite practical to build a multiple-ported asynchronous communications adapter with onboard character and modem control signal protocol processing. Such an asynchronous adapter would significantly reduce the system overhead presently associated with serial port (RS-232C/RS-422) support.

The important point in these examples is to show how the bus master capability of Micro Channel architecture opens many new possibilities for achieving parallelism between I/O-driven operations and other system functions in the multi-tasking environment. With these examples as background, let's look now at the OS/2 services-enabling bus master support.

OS/2 Device Driver Model

We have already indicated the importance of the OS/2 device driver model. The reason is that hardware interrupts are essential to the efficient asynchronous operation of a bus master. In OS/2 a device driver is the only component of the system that can register to receive control when a device issues a hardware interrupt.

The system does not do a full-process context switch before calling a hardware interrupt handler. Staying

within the context of the active process minimizes interrupt latency (the time it takes to get to the correct interrupt handler when a hardware interrupt occurs). This also reduces the overall system interrupt-handling overhead. On the other hand, running in an unknown process context imposes significant restrictions on the interrupt handler. Performing device I/O outside the context of the requesting thread is one of the primary programming considerations addressed within the OS/2 device driver model.

Besides containing interrupt handlers, device drivers are the most

privileged user-installable components of the system. The OS/2 architecture treats a device driver as an extension of the operating system kernel. This includes making available many system services that are not available to normal application software. Several of these services are particularly necessary to support a bus master device. From a structural and system service standpoint, a subsystem supporting a bus master in OS/2 must incorporate a device driver component.

Bus Master Configuration and Initialization

OS/2, as with PC DOS, does not get involved in selecting or initializing the Programmable Option Select (POS) values for a bus master adapter. The system hardware SETUP program and the BIOS Power-On Self-Test (POST) provide these functions. If the adapter supports a base initial program load (IPL) device, it has additional initialization constraints. It should have no dependency on loading a device driver in order to support execution of the SETUP program or the early stages of operating system initialization.

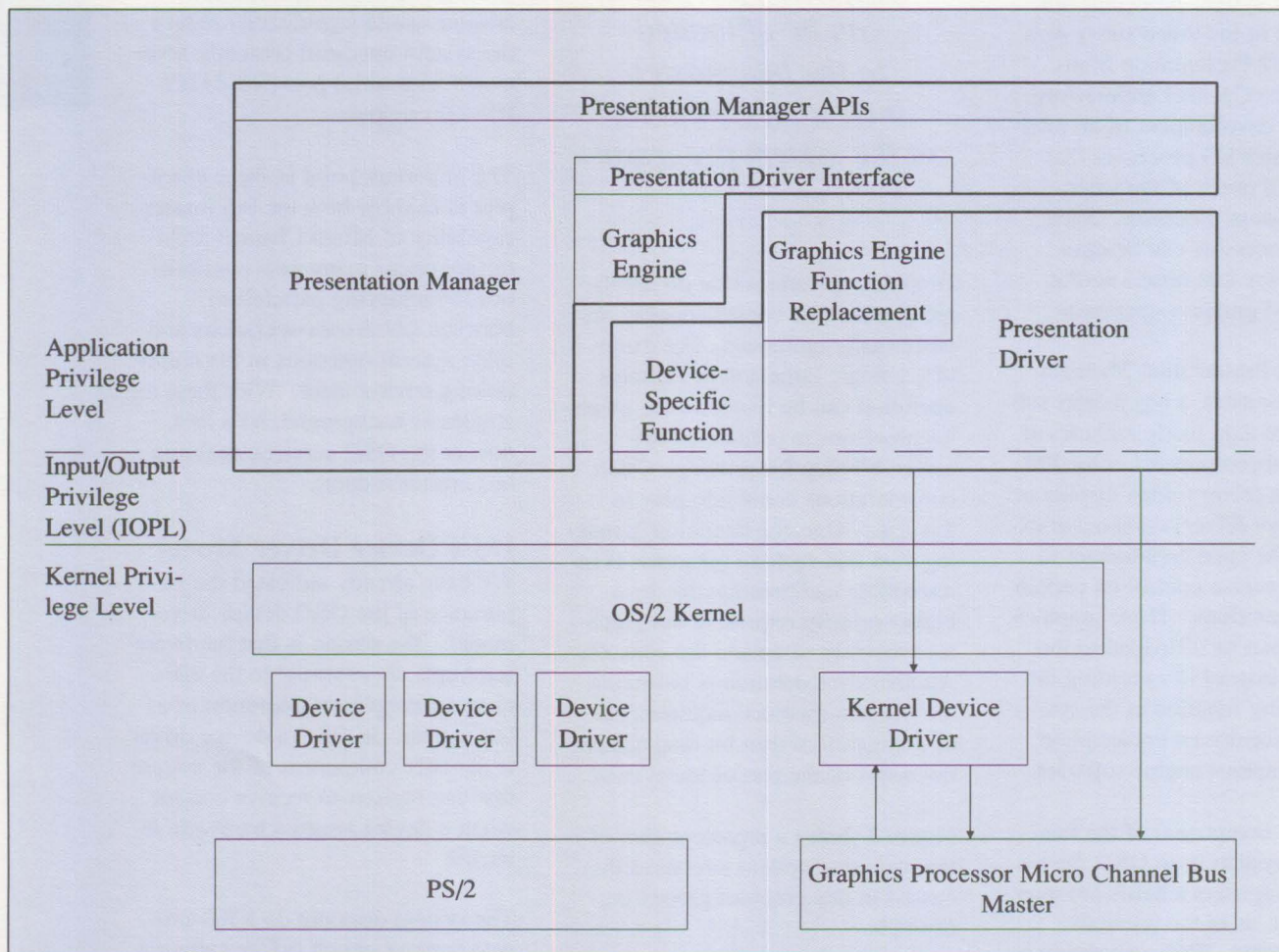


Figure 3. OS/2 Bus Master Graphics Processor Subsystem

Bus master adapters for these devices should have sufficient logic and/or some form of non-volatile memory (that is, ROM) to operate in a minimum function mode until a device driver can be loaded. For example, a DASD device should power-up with support for the BIOS real mode INT 13 functions if it is to contain a bootable volume. Similarly, the primary console video adapter should power-up in some default VGA-compatible mode. Once system initialization progresses far enough to load an installable device driver, a bus master adapter subsystem can enable more advanced function modes.

Upon initialization, the device driver for a bus master adapter is responsible for determining any configurable values that are set by the POS mechanisms to initialize the adapter. There are several techniques that can be used to accomplish this. One approach is to find the POS registers for the adapter and then read and interpret the POS register values. A device driver may have little use for some of the POS information, such as the bus arbitration level. Other parameters that must be understood by the device driver include the physical addresses for any memory-mapped buffers, any I/O port addresses, and the hardware interrupt level(s) used by the adapter.

OS/2 System Services Important to Bus Master Device Drivers

Two volumes of the *IBM Operating System/2 Technical Reference*, titled *I/O Subsystems and Device Support*, contain a complete description of the programming model and services for OS/2 device drivers. Volume 1 addresses the privileged device drivers that effectively ex-

tend the operating system kernel. Volume 2 addresses presentation drivers that support the graphics capabilities of Presentation Manager.

A complete description of all the services available to an OS/2 device driver is beyond the scope of this article. Let's concentrate on pointing out some of the specific services that can be used to support the bus master scenarios we have discussed.

The level-sensitive interrupts of the Micro Channel bus are well suited for interrupt sharing.

The special system services available to OS/2 device drivers are called Device Helper Services, or DevHelps. These DevHelp services are the functions described in the following paragraphs.

Registering for an Interrupt

Level: The function **SetIRQ** is used to register an entry point to receive control on a hardware interrupt. The level-sensitive interrupts of the Micro Channel bus are well suited for interrupt sharing. A bus master device driver can specify whether it is willing to share its interrupt level when registering for it with the operating system. The advantage of a bus master sharing its interrupt level is that it could potentially coexist with more interrupting devices in the system, including multiple instances of itself. The disadvantage is that there is a performance overhead associated with

polling the devices on a shared level to see which one should respond to an interrupt event.

Memory Management Services:

There are a number of memory management functions available to support the addressing requirements of a bus master device driver. The virtual memory support of the Intel 286 architecture does not affect the single I/O address space. In other words, all processes in the system share the same I/O port address space. Therefore, no services are necessary to support any type of I/O port address translation. The main memory address space, on the other hand, has a number of considerations. First, the virtual addresses used in software for execution by the system processor are different from the physical addresses used across the Micro Channel bus. Second, these addresses are process-context-sensitive and are not usable outside the context of the I/O thread such as at interrupt time. Finally, the addresses are specific to the real and protected addressing modes of the 286 processor.

There are different services to translate a physical address to a virtual address, depending upon how the address is to be used. If a bus master device driver wants to establish global addressability (not context-sensitive) to a memory-mapped buffer, it can use the function **PhysToGDTSelector**. This function should be used sparingly because the Global Descriptor Table (GDT) is a limited resource. Before using this function, the device driver must have first allocated a GDT entry using the function **AllocGDTSector**. The address returned by **PhysToGDTSelector** is a privileged address that is usable only within the device driver. If the device driver wants to pass a bus mas-

ter buffer address back to some user (application privilege level) software for its use, it would use the **PhysToUVirt** function. The addresses created by this function are within the user privilege domain, and only valid within the context of the process owning the thread that issued the **PhysToUVirt** function call.

The device driver may want to convert a virtual address passed from a user-level program to a physical address. A physical address must be used in a Micro Channel DMA operation (first-party bus master or third-party DMA). The function for this type of address conversion is called **VirtToPhys**. Because there is only one physical address space in a PS/2 system, this function is also used to obtain a context-free normalized form of a virtual address. This

"normalized" physical address is then usable outside the context of the thread requesting the I/O, such as at interrupt time, to call the function **PhysToVirt** and obtain a temporary virtual pointer for the address.

Virtual pointers, not physical pointers, must be used for memory references in instructions executed by the system processor. The virtual address created by **PhysToVirt** is valid for use within the device driver because it is allocated at the operating-system level of privilege. Note that there is one necessary step before a virtual address can be converted to a physical address that will remain valid over time. In order for the physical address to remain valid, the memory object pointed to by the virtual address

must be locked in physical memory using the function **Lock**.

Another feature of the **PhysToVirt** function is that the virtual address returned is guaranteed to be valid for the current addressing mode (real mode versus protected mode) of the 286 processor. This bimodal support is a significant part of the device driver I/O support of the OS/2 PC DOS application compatibility environment. OS/2 device drivers can be written so that regions of code can execute whether the processor is in real or protected mode.

On the other hand, a device driver may choose not to execute in real mode. For example, if a bus master device driver finds that its interrupt handler has gained control while the processor is in real mode, it can use the functions **RealToProt** and **ProtToReal** to get to and then return from protected mode. This could be critical if the interrupt handler must address some memory object located above the 1 MB address boundary of real mode.

Synchronizing Primitives and Process Management:

There are two basic approaches to thread synchronization that a device driver can use. One is to use the explicit **Block** and **Run** functions, which employ an arbitrary and yet global 32-bit event ID value. There is no registration for this value. The convention used to avoid conflicts is to use the physical address of the I/O request packet associated with the blocked thread for the event ID value. A slightly safer synchronizing approach is to use a system semaphore and its associated functions **SemHandle**, **SemRequest** and **SemClear**. Although there is slightly more overhead associated with their use, system semaphores

Interrupt Management	
SetIRQ	
UnSetIRQ	
Memory Management	
AllocGDTSelector	
PhysToGDTSelector	
PhysToUVirt	
PhysToVirt (UnPhysToVirt)	
VirtToPhys	
Lock (Unlock)	
Addressing Mode Management	
RealToProt	
ProtToReal	
Synchronization Primitives and Process Management	
Block	
Run	
SemHandle	
SemRequest	
SemClear	
Yield	
TCYield	

Figure 4. Services Important for Bus Masters

are guaranteed to be unique. A system semaphore will also be released if the owner of the semaphore terminates.

One of the characteristics of the OS/2 device driver model is that a thread executing down in a device driver will not be preempted (except by a hardware interrupt). Therefore, it is the device driver's responsibility not to execute for extended periods without yielding the system processor. It is particularly important for the device driver to yield to any higher priority, time-critical thread that may have transitioned to the ready-to-run state. There is a system data structure (yield flag) that a device driver can check to see whether some other thread is scheduled to run. The functions available to a device driver to release the system processor are **Yield** and **TCYield** (yield only to a time-critical thread).

These are just some of the OS/2 system services available that should be of particular interest to developers of bus master device drivers. Most of the services described have

corresponding services for undoing or deregistering the respective service. There are many other services not discussed in this article that will be useful to the bus master subsystem developer. These services include functions for allocating physical system memory for device-driver data structures and buffers, managing I/O request packet queues, and requesting timer services.

The services discussed in this article are summarized in Figure 4.

Summary

We have discussed how the multithreaded programming model enabled by a multitasking operating system such as OS/2 creates an environment that the bus master capability of the Micro Channel architecture is ideally positioned to support. We have reviewed some of the considerations for exploiting the structures and services of OS/2 to build a bus master-based, intelligent I/O subsystem. It is clear that Micro Channel architecture is an excellent platform for the future which is, in fact, already here for multitask-

ing subsystem developers. Opportunities for a new generation of performance and function in the PC-compatible market are limited only by the imagination of Micro Channel bus master subsystem builders.

ABOUT THE AUTHOR

Robert L. Williams is currently a software development manager in the OS/2 Control Program development organization, and was previously involved with the design of OS/2 as a member of the OS/2 Systems Architecture and Design team. He joined IBM's Entry Systems Division in 1983 and has worked on several personal computer office system software projects in areas including telephony, LAN communications, and document retrieval. Bob is coauthor of a book titled OS/2 Features, Functions and Applications (John Wiley & Sons, Inc.). He holds bachelor's and master's degrees in computer science and engineering from the University of South Florida.

Micro Channel Issues in AIX PS/2

Robert Mercier, Richard Bass,
and Jill Dinse
Locus Computing Corporation
Inglewood, California

This article describes the features of the Micro Channel system as they relate to the implementation of Advanced Interactive Executive (AIX) on the PS/2 computer. The purpose of any operating system is to bridge the gap between the user and the hardware. While most of the AIX operating system is portable, there are portions that are hardware-specific. Handlers for Micro Channel devices, device drivers, are examples of such machine-specific code. It is these device drivers that communicate with devices and, depending on the device, may take advantage of the facilities of the Micro Channel system.

Micro Channel architecture contains several features that allow efficient utilization of the resources of the PS/2 system. We will discuss several of these features as they relate to the implementation of AIX on the PS/2. Topics include the Programmable

Option Select, DMA, and interrupt chaining. Authors of device drivers for AIX may find this information particularly useful.

The advent of Micro Channel architecture has brought about a new level of technology for the personal computer user. The facilities it provides, along with the capabilities of the Intel 80386 microprocessor, create a strong platform on which to support advanced operating systems such as AIX. This article describes the Micro Channel-related aspects of AIX on the 80386-based PS/2.

It is the duty of the operating system to maximize the use of the hardware by users of the machine. Recently, it has become generally accepted that the operating system should mask the details of the hardware from the user and provide a general, consistent interface to devices. This philosophy is most strongly embodied in the UNIX operating system, on which AIX is based. The ease with which this task can be accomplished is a characteristic of the machine architecture. The PS/2 line of computers, using the 80386 microprocessor and the Micro Channel architecture, provides a good foundation on which to implement multiprocessing, multi-user operating systems.

AIX Device Interface

In this section, we give a general overview of the kernel/device interface in the AIX kernel. Readers familiar with these aspects of AIX or UNIX may continue to the next section.

Figure 1 provides a view of the relationship between the user, the kernel, and the devices. The device interface between applications and the kernel is through a special file system entry. These files are referred to as device special files and are commonly grouped together in the /dev directory. These files possess two attributes, major and minor numbers, that establish their relationship with specific device drivers. The major number ties the device special file to a specific driver and the minor number denotes a subunit of a particular device.

Common I/O operations on device special files such as open, close, read, write, and so on, are routed through a table of indirect function pointers stored in the kernel (devsw in Figure 1). The major number is used as the index to this table. The minor number is passed as an argument to the appropriate routine.

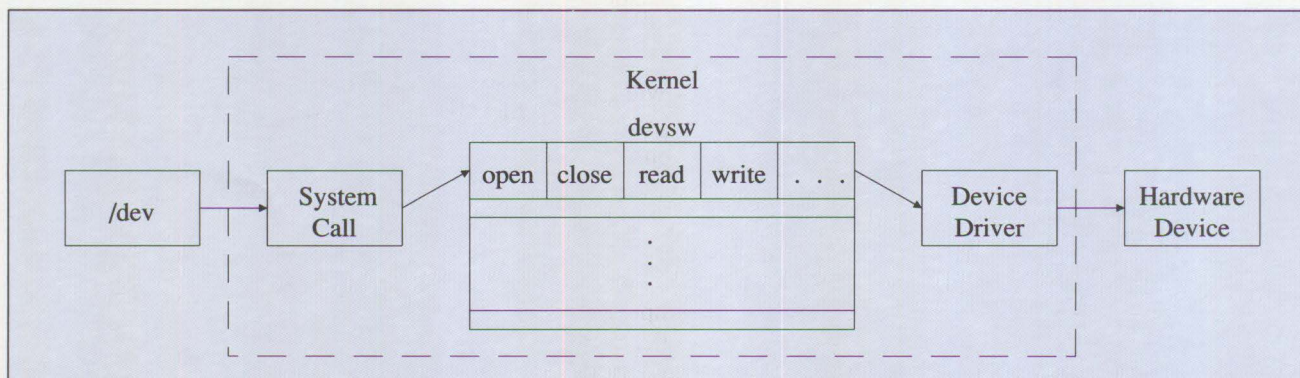


Figure 1. Relationship Between User, Kernel, and Devices

```

#define TKRPSID    0xE000  /* PS/2 pos cardid */
/* return the next slot for the token ring */
for (slot = 0; (slot = devexist(TKRPSID, maj, 1, slot)) = 0; slot++) {           /*
.
.
.
}

```

Figure 2. Driver Initialization Routine

When drivers initialize, they typically advertise a routine that they want to have called when an interrupt occurs at a specified interrupt request (IRQ) level. When a user requests I/O to a device, and the operation cannot be completed immediately, the operation is initiated and the process puts itself to sleep, enabling other processes to run. When the operation completes, the device interrupts the system unit, and the device-driver interrupt routine is called, the interrupt routine marks the operation as complete and the kernel wakes up the sleeping process. If the interrupt routine finds that there is more work to do, another operation is initiated and the entire process is repeated.

Driver Types: There are essentially two types of drivers. Those which transfer data between the kernel and the device in small units, typically single characters, are referred to as character devices. Examples are printers, serial line adapters, and mouse drivers. When the device is capable of larger, typically fixed-size transfers and can provide random access to some external storage, then it fits the block device paradigm. This group includes diskette and fixed-disk drivers and some tape drivers. It is the responsibility of character device drivers to make it appear to higher levels of the kernel that data received from a device resemble a linear stream of characters. For block

device drivers, the data should appear to be a randomly accessible collection of fixed-size blocks numbered consecutively beginning with zero to end of the media.

In the following sections, we will discuss the features specific to the Micro Channel system and their relationship to the implementation of AIX device drivers on the PS/2.

Programmable Option Select

One of the major frustrations experienced by users of many machine architectures is the installation and setup of device adapters. Previous designs typically required that users set a variety of non-intuitive switches and jumpers on the adapter before installing it in the machine. Results of incorrect switch settings range from mystifying errors to hardware damage. Users were required to read and understand complex installation instructions and be familiar with the handling of static-sensitive equipment.

The PS/2 line of computers, with Micro Channel architecture, provides an adapter setup and configuration mechanism managed completely under software control. Every adapter designed for the Micro Channel system must adhere to this mechanism. PS/2 computers are shipped with a configuration program, and adapters are accompanied by Adapter Description Files de-

scribing the resources used by the adapter. This facility is collectively referred to as the Programmable Option Select (POS).

User Setup: Each adapter is required to supply a 16-bit card identification (ID) number when queried by the system unit. This number indicates the function of the adapter. For example 0xEEFF has been reserved for serial adapters. In all, 8 bytes are available for each adapter to communicate option settings to and from the system unit. Certain fields are reserved for specific functions such as DMA arbitration level, device ROM segment address, device RAM segment address, and device I/O address. Four of the bytes are completely free-form, and their use is defined by the adapter manufacturer.

The configuration program, using the Adapter Description File, allows the user to modify these settings in an interactive, menu-oriented way. For reserved fields, the program also ensures that there are no conflicts between adapters.

Device-Driver Interface: During kernel initialization, each slot is queried for its card ID and configuration information. As each driver initializes, it calls a utility routine to determine whether its corresponding adapter is present. The example in Figure 2 contains an excerpt from the IBM Token-Ring driver initial-

```

struct devdata {
    unsigned char    pd_pos0; /* Card ID low */
    unsigned char    pd_pos1; /* Card ID high */
    unsigned char    pd_pos2; /* option select data byte 1 */
    unsigned char    pd_pos3; /* option select data byte 2 */
    unsigned char    pd_pos4; /* option select data byte 3 */
    unsigned char    pd_pos5; /* option select data byte 4 */
    unsigned char    pd_pos6; /* subaddress extension LSB */
    unsigned char    pd_pos7; /* subaddress extension MSB */
    unsigned short   pd_major; /* major dev */
    unsigned short   pd_flags; /* flags */
};
struct devdata devdata[NSLOTS];

```

Figure 3. Global Array Describing Adapter Slots

ization routine. The function **devexist** takes four parameters: the adapter identification number, the device major number, flags, and a slot number. The adapter ID is used to locate the adapter by type. If an adapter with this ID is located, the major number and flags are stored in the kernel structure that describes the slot containing the adapter. The slot number indicates the position at which **devexist** should begin searching. This allows **devexist** to locate multiple cards with the same ID.

devexist returns an index to a global array that describes the slot containing the adapter (Figure 3).

The driver can use the information in this structure to determine how to communicate with the adapter. For example, the Ethernet adapter supports a configurable read-only memory (ROM) address window. The Adapter Description File indicates which bits in the POS information correspond to which ROM addresses, and the piece of code in Figure 4, taken from the driver, decodes the information.

The Adapter Description File must advertise the meaning of these bits so that the configuration program can prevent conflicts.

Additional information about the Programmable Option Select, including the format of the Adapter De-

scription Files, is available in the *IBM Personal System/2 Hardware Interface Technical Reference* manual.

Shared Interrupts

A significant feature of Micro Channel architecture is the use of a level-sensitive interrupt mechanism. This feature allows multiple adapters to interrupt the system unit at the same level simultaneously. Adapters are required to provide an interrupt-pending latch readable by the system unit so that drivers can determine whether their device is responsible for the interrupt.

In AIX, a driver "signs up" for the interrupt levels it is interested in examining. For devices supporting

```

seg = (unsigned int) devdata[slot].pd_pos4 & 0x0f;
off = 0x4000 * (unsigned int)
((devdata[slot].pd_pos5 > 2) & 03);
rom_addr = (unsigned int) (seg < 16) | off;

```

Figure 4. Decoding Adapter Description File Information

```

intrattach(saintr, 3, SPL_HIGH);
intrattach(saintr, 4, SPL_HIGH);

```

Figure 5. Example of intrattach Routine

```

struct intrshare {
    int (*intr_handler)(); /* The interrupt handler */
    /* Pointer to the next handler */
    /* for this interrupt level. */
    struct intrshare *intr_next;
    int intr_plevel;      /* Requested spl level */
};

/* Interrupt vector switch */
struct intrshare *intvecsw[NUM_INTVECS];

```

Figure 6. Format of Array of List of Drivers

configurable interrupt levels, this information is typically supplied as part of the Programmable Option Select. During device initialization, each driver registers an interrupt service routine by calling the **intrattach** routine. Figure 5 illustrates this.

The arguments indicate the routine to be called, the interrupt level, and the priority-level mask, respectively. This mask is used to selectively disable interrupts when the interrupt handler is called. All drivers for devices of a common type use the same mask value. The example above is from the serial device driver. All such drivers use **SPL_HIGH**. Similarly, block device drivers, such as fixed-disk drivers, use **SPL_BLKIO**. This mechanism helps provide consistent system processor utilization and priority to devices requiring prompt attention.

When the **intrattach** routine is called, the service routine and mask

are placed on a list of drivers interested in the specified interrupt level. The kernel maintains an array of such lists, indexed by interrupt-level number. The format of the array is shown in Figure 6.

At kernel initialization time, all hardware-interrupt vectors are pointed at a single global interrupt-handling routine. When an interrupt occurs, the global handler resets the interrupt controller (End Of Interrupt), indexes into **intvecsw** by interrupt-level number, traverses the **intr_next** list, masks out all interrupts below **intr_plevel**, and then calls **intr_handler**.

Each interrupt handler is responsible for determining whether its device caused the interrupt. An interrupt-pending latch is required of each adapter for this purpose. If a driver's interrupt routine is called, and the driver finds that its adapter is not asserting an interrupt, then it simply returns. Figure 7 contains

the prologue of a representative device interrupt handler (the 6157 Streaming Tape Adapter).

The **inb** function reads a byte from an I/O address, in this case, the Status register for the tape adapter. This is an active low flag on the tape adapter, so the above example is checking for a 0 bit. If several devices were sharing the same interrupt level and an interrupt occurred, all but one would return immediately when their interrupt handlers were called.

The interrupt-level sharing, combined with the facilities provided by Programmable Option Select, results in an extremely flexible configuration mechanism easily manipulated by users with modest technical expertise.

DMA Facilities

The Micro Channel DMA controller contains several features that facilitate easy and efficient support of a wide range of device controllers.

```

if ((inb(MTFBP1R) & INTTR) || (! mtintrenabled))
return; /* not for us */
/* handle interrupt */
.
.
.

```

Figure 7. Prolog of a Device Interrupt Handler

```

struct dmaralloc {
    /* Driver initialized elements */
    char *dma_devicename;    /* Name of the device (as an ASCII string) */
    int (*dma_availfunc)(); /* Function to call when resource available */
    short dma_priority;      /* Relative priority */
    short dma_arblevel;     /* Requested arbitration level */

    /* dma.c initialized elements */
    short dma_channel;      /* Channel assigned by dmachanalloc() */
    short dma_flags;       /* Flags */
    struct dmaralloc *dma_next; /* Next pointer of linked list */
};

```

Figure 8. DMA Resource Allocation Structure

These include register compatibility with the AT bus 8237 controller, eight independent channels, two programmable channels, and selectable byte or word transfer modes.

The most interesting feature of the Micro Channel DMA controller is the two programmable DMA channels. These channels can be set to any arbitration level. With a little careful programming, it is possible to support multiple devices at the same arbitration level, a feature normally disallowed by the Micro Channel system. While devices sharing the same arbitration level experience some performance loss, the ability to support them is important.

DMA devices communicate with the system unit DMA controller over one of 15 arbitration levels. For levels 1 through 3 and 5 through 7, the device uses the corre-

sponding DMA channel, because these channels are hardwired to the specified arbitration level. By this we mean that a device set to arbitration level 2 would use DMA channel 2 because it is permanently set to arbitration level 2. Devices set to levels 0 or 4 or levels 8 through 14 all share programmable channels 0 and 4.

While these restrictions may seem to limit the AIX kernel, there is an important positive aspect. DMA channel-allocation routines allow drivers to allocate channels at the arbitration levels of their devices without concern that another adapter might be set to the same arbitration level.

DMA Channel Allocation: Each driver that intends to use the DMA controller must provide a DMA resource allocation structure (Figure 8).

The driver initializes the first four fields before calling **dmachanalloc**. The fragment in Figure 9 is taken from the enhanced small device interface (ESDI) disk driver.

The **dma_devicename** field is informational. The **dma_availfunc** is part of a call-back mechanism for managing multiple, simultaneous requests for the same arbitration level. If the driver attempts to allocate a channel for some arbitration level, and the request cannot be satisfied because the resource is busy, the allocation fails. When a suitable channel is later freed, the **dma_availfunc** is called.

Many devices manage interrupt-driven I/O through two cooperating functions commonly known as **start** and **intr**. When an I/O operation is requested, it is typically placed on a

```

struct dmaralloc ehddma = {
    "ESDI hard disk", /* device name */
    ehdstart, /* dma_avail func (called when chan becomes avail) */
    DMA_AVEPRI, /* priority */
    0 /* arb level to be filled in by ehdinita() */
};

```

Figure 9. Fragment from ESDI Disk Driver


```

/*
 * Try to allocate a DMA channel. If we can't, dmachanalloc()
 * will set up to call us (ehdstart()) again when the right
 * channel becomes available.
 */
if (( !(ehddma.dma_flags & DMA_HAVECHAN)) &&
    ( !dmachanalloc(&ehddma)))
    return;

```

Figure 10. Example from ESDI Disk Driver Start Routine

queue of pending operations. If the driver is idle, the **start** routine is called to remove the first item from the queue and initiate the operation. The device will generate an interrupt when the I/O operation completes and **intr** will be called. Before it returns, it will check the pending queue and call **start** again if there is more work to do.

By carefully coding **start**, it is possible to arrange for the DMA allocation routine to use **start** as the **dma_availfunc** or call-back routine. The example in Figure 10, from the ESDI disk driver start routine, illustrates this.

If the driver does not already have the channel and is unable to allocate it, then return. When the **dmachanalloc** failed, the **dmarralloc** structure was placed on a list of drivers waiting for this channel. The channel lists are kept sorted by **dma_priority**. All requests requiring programmable channels are placed on the list for channel 0. When a channel is

freed, the head of the list for that channel is removed and its **dma_availfunc** is called. Care is taken so that when either of the programmable channels is freed, the list for channel 0 is used. In the example above, if the channel was initially busy, **ehdstart** would be called by the DMA channel free routine when a suitable channel became available.

DMA Usage: A function called **dmasetup** is provided to set up a variety of DMA transfers. It can initiate transfers through I/O space or by using arbitration levels, and it allows the selection of 8- or 16-bit transfer sizes. Figure 11 contains an example from the ESDI disk driver:

The arguments are the physical address of the buffer, the function to perform (**B_READ** if read operation, otherwise write is assumed), the length of the transfer, a pointer to the **dmarralloc** structure, the I/O space address to program into the controller, and the transfer size in bytes. The **outb** statement instructs

the ESDI controller to initiate the operation.

When the DMA operation completes, the channel should be freed even if there is more work pending. Even though the interrupt routine will call **start** if there is more work, and it seems as if the channel will become available immediately, there are cases in which this will not happen. It is **dmachanfree** that calls the **dma_availfunc** functions so that if another driver is waiting to use this channel, it will be allocated as soon as it is free. Drivers waiting to use channels are serviced round-robin when priority levels are equal.

The DMA channel-management facilities and drivers written to use them help to ensure that nearly every possible combination of arbitration-level requests can be satisfied. This design is in keeping with the philosophy of Micro Channel technology: increased flexibility and function.

```

dmasetup(bp-b_physaddr, bp-b_flags & B_READ, bp-b_bcount,
    &ehddma, EHDDMAIO, sizeof(unsigned short));
outb(BCR, BC_DMAENABLE | BC_INTRENABLE);

```

Figure 11. Example of Transfers from ESDI Disk Driver

Conclusion

The Micro Channel system provides a rich environment for supporting a wide variety of devices in an efficient and flexible manner. The Programmable Option Select feature combines well with the programmability of the DMA controller and the ability of the interrupt controller to share levels. When adapters are constructed and drivers are written to take advantage of these features, machine setup and configuration become easy enough for even the novice user.

The AIX interface to these resources was designed to facilitate the easy addition of new drivers to the operating system while maintaining efficient utilization of the hardware. In particular, the ability of the DMA allocation routines to support multiple adapters at the same arbitration level allows the use of devices where it might be otherwise impossible.

Additional information about these topics, as well as a comprehensive guide to writing device drivers for AIX can be found in "Writing Device Drivers," *AIX PS/2 Technical Reference*, Volume 2, Appendix C.

ABOUT THE AUTHORS

Robert Mercier received his bachelor of science degree in computer science from Indiana State University and has been at Locus Computing Corporation for two years. He was part of the kernel development team for AIX PS/2 release 1, concentrating especially in the Micro Channel aspects of the operating system. He is currently working on an X-windows development project.

Richard Bass received his bachelor of science degree in information and computer science from the University of California at Irvine and has been at Locus Computing

Corporation for four years. He has worked on various projects at Locus, including operating system software for personal computers. Mr. Bass was a key technical lead in the AIX PS/2 release 1 project, particularly in the Micro Channel area. Currently, he works as a technical lead in the AIX PS/2 release 1 Level 4 support team.

Jill Dinse received her bachelor of arts degree in professional writing from Carnegie-Mellon University and has been at Locus Computing Corporation for two years. She has worked on various projects at Locus, including documentation for AIX PS/2 release 1. Currently, she is manager of the Custom Division Technical Publications Department and is working on documentation for AIX PS/2 release 2 and AIX/370 release 1.

Information for Developers

Mili Sanderson
IBM Corporation
Boca Raton, Florida

Often, as with any task, getting the necessary information in order to get started is a major stumbling block. This article gives step-by-step information on obtaining necessary information about developing products for Micro Channel architecture, as well as the steps IBM has taken to assist developers in publicizing their products for Micro Channel systems.

Technical Reference Library

A technical reference library is available. It contains publications providing PS/2 system-specific hardware interface information for developers of products that operate with the PS/2 systems. Information is presented in three major library sections covering BIOS interface, systems, and options and adapters.

A complete listing of these publications, part numbers and pricing may be obtained by calling:

1-800-426-7282

and requesting a *Technical Directory*. There is no charge for this booklet.

Micro Channel architecture interface and design information for the various PS/2 system units, including the system board, co-processor, power supply, keyboard, instruction

sets, characters, keystrokes and colors, and other features of the system is available in the "System" section of the *IBM Personal System/2 Hardware Interface Technical Reference* manual, part number 68X2330. This technical reference manual is for the PS/2 Models 50, 60, 70 and 80 computers. It may be ordered through your nearest IBM branch office library. If you don't know the location of your nearest IBM branch office, you may call:

1-800-426-3333

for the location and phone number for the nearest IBM branch office.

A supplement to this technical reference manual is also available for the PS/2 Model 70-A21. It is part number 68X2344.

Programmable Option Select (POS) Number

Switch settings are an encumbrance of the past when developing an adapter for IBM PS/2 Micro Channel computers!

By including an assigned POS number (Micro Channel Card ID) on the option setup diskette accompanying a new adapter, the user simply copies this information onto the PS/2 Reference Diskette during installation of the adapter, and the system will automatically recognize the adapter.

To obtain a Micro Channel Card ID (POS number) for your adapter, call:

1-800-426-7763.

For Canadian callers, the following extensions are available:

1-404-988-1515/1878/1879.

Your call will be answered by a bank of switchboard operators at the IBM National Support Center. Because these operators handle a high volume of calls on various subjects each month, it is very important to indicate that you are calling to obtain a Micro Channel Card ID. Your name, company name and telephone number will be taken and routed to a Card ID representative who will return your call as quickly as possible.

Card IDs are assigned from five different numeric hexadecimal sequences based on the type of Micro Channel adapter to be developed, as follows:

0000 Series = Bus Master

Adapter: A bus master adapter is capable of taking control of and managing all activities of data transfers. Micro Channel architecture permits multiple bus master adapters to exist at once in a PS/2 system and to function in an orderly fashion.

5000 Series = Direct Memory

Access (DMA) Adapter: A DMA adapter is a slave adapter. It works under control of either the DMA controller on the PS/2 system board or a DMA controller on a bus master adapter. It permits transfers of information between system input/output (I/O) space and system memory directly, without processor intervention.

6000 Series = Direct Program

Control (DPC) Adapter: A DPC adapter is also a slave adapter. It sends and receives data under control of either the system logic or a bus master adapter. A DPC adapter contains I/O ports within the system I/O space, which are accessible by any other adapter.

7000 Series = Memory Adapter:

A memory adapter is also a slave adapter. It sends and receives data under control of either the system logic or a bus master adapter. It contains memory modules within the system memory space, which are accessible by any other adapter.

8000 Series = Video Adapter:

A video adapter is also a slave adapter. It is a specialized type of DPC adapter that handles the transfer of data between the Micro Channel bus and a video peripheral device.

The Card ID representative will discuss your requirement with you, obtain an appropriate ID, assign it to your company, and confirm the assignment of this number to you by mail within three days.

IBM Developer Assistance Program

Through this program, IBM provides technical information and assistance to developers to help them design and develop software and hardware products for IBM DOS and IBM OS/2 operating systems on IBM PC or PS/2 computers.

To become a member of the IBM Developer Assistance Program, call:

1-407-982-6178.

You will be mailed a four-page application to complete and return.



If you wish, you may request this application by writing to:

IBM Corporation
Developer Support - 4607
P.O. Box 1328
Boca Raton, FL 33429-1328

Eligibility is limited to U.S. companies or U.S. subsidiaries of non-U.S. companies who are currently marketing products for commercial release and who are not already IBM Industry Remarketers. If you are not yet marketing your products, you may submit a non-confidential detailed business plan showing marketing and development activities and schedules.

Benefits of becoming a member of the IBM Developer Assistance Program include the following:

- Invitation to advanced technical education classes conducted by IBM at a variety of locations throughout the United States. These classes provide the opportunity for you to enhance your development skills.

- Technical mailings that provide current information directly from IBM to you. This information is provided in the form of seminar proceedings, technical bulletins and a quarterly publication called the *IBM Personal Systems Developer*.

- Participation in discounts/rebates on selected software and hardware products for recognized developers who are committed to developing products for IBM OS/2 and IBM DOS operating systems.

- Advertising assistance for your qualified products by publishing your product information in either the *IBM OS/2 Application Guide* or the *Catalog of Micro Channel Hardware Products*.

After you are accepted into the IBM Developer Assistance Program, you will be sent a confirmation package, including information on how to obtain technical support, and specific information on the discount pro-

grams and current technical information available to developers.

Merchandising Assistance

Catalog of Micro Channel

Hardware Products: Since Fall/COMDEX in November 1987, IBM has continued to publicize those developers of Micro Channel hardware and their products in the IBM booth at COMDEX. The original list of developers with their product names has been expanded through 1988 and into 1989 by the formal publication of the *Catalog of Micro Channel Hardware Products* offered by Independent Option Vendors (IOVs) as well as by the IBM Corporation.

The third edition of this catalog was released April 10, 1989, at Spring/COMDEX '89. The fourth edition is planned for release at Fall/COMDEX '89 on November 13 in Las Vegas, Nevada.

To date, over 120,000 copies of this catalog have been made available worldwide to dealers, marketing representatives, Industry Remarketers, corporations and end users. Participating developers have reported increased interest in their products based on the information published in this catalog.

An independent survey of the developer information published in this catalog indicated that the information presented was an unprecedented 95.3 percent accurate.

If you have obtained a Micro Channel Card ID from IBM, your company will be automatically contacted to participate with us in having your product(s) for the Micro Channel-based PS/2 computers listed in this catalog. If your

product is not currently enrolled with us, and you would like to participate in this program, you may call:

1-407-982-5361

to request a Hardware Product Enrollment form. You may also request this form by writing to:

Ms. Mildred N. Sanderson
IBM Corporation - 1817
P.O. Box 1328
Boca Raton, FL 33429-1328

The Micro Channel products listed in this catalog fall into the following 36 major categories:

- Audio/Voice Phone Products
- Cables
- Co-Processors
- Chip Sets
- Data Acquisition Products
- Debug Tools
- Diskette Drives
- Displays & Display Adapters
- Encryption Devices
- Facsimile Products
- Fixed Disk Drives
- Game Ports
- Host Communication Products
- Image Capturing Products
- Keyboards
- Light Pens
- Memory Products -- 16-Bit
- Memory Products -- 32-Bit
- Miscellaneous Accessories
- Miscellaneous Adapters
- Modems
- Mouse Products
- Network Adapters
- Optical Disk Drives
- Parallel Port Adapters
- Performance Products
- Plotters
- Printers
- Programming Tools
- Scanners
- SCSI I/O Attachments
- Serial Port Adapters
- Serial/Parallel Combinations
- Tape Drives
- Terminal Emulators
- Video Projection Products

Bus master adapters listed under one of these category types will also be listed under an additional "Bus Master Adapters" category in the next edition of the catalog.

Wall of Cards: In Spring 1988, IBM displayed 84 adapters for Micro Channel computers from Independent Option Vendors at COMDEX. This "wall of cards" was expanded for Fall/COMDEX that year to include 234 IOV adapters for the Micro Channel computers. The attention attracted by these two displays of available products for the PS/2 computers received national publicity.

The catalog and the "wall of cards" were also made available to an extremely interested international audience at business shows in Tokyo and Osaka, at PC Forum in France and at the Hanover Faire in Germany, as well as at PC EXPO shows.

The number of available adapters has now become so large that display and transport of any "wall of cards" has become prohibitive. Therefore, at Spring/COMDEX '89, a "wall of bus master cards" was displayed along with an online demonstration of all of the products available for the Micro Channel-based PS/2 computers.

Electronic Catalog: In December 1988, the contents of the catalog was made available electronically to anyone worldwide who has a PC or PS/2 computer, a modem and communications software. The data bases of IBM and IOV hardware products for the Micro Channel computers were added to the IBM PC User Group Support Bulletin Board System (PCUG BBS) sponsored by the IBM National Support

Center in Atlanta, Georgia. These data bases are updated monthly.

Access to the IBM PC User Group Support Bulletin Board System can be obtained by calling:

1-404-988-2790.

The only cost of access to this system is that of the phone call. Registration consists solely of your name, a password you assign, your baud rate, the location you are calling from and a daytime phone number.

Once you are into the PCUG BBS main command screen, the Micro Channel hardware products are accessed by typing "OPEN 3". Other information available through the PCUG BBS includes bulletins on the latest IBM product announcements, a PC User Group locator data base, OS/2 operating system compatibility application data base, message sending and receiving, online chat and numerous files, including fixes and utilities, that can be downloaded.

The PCUG BBS is set up with a PS/2 Model 55 SX computer with a 120 MB fixed disk as the dedicated server. At the end of May 1989, there were a total of nine nodes available for callers. Five of these were PS/2 Model 60 computers sharing their fixed disks to provide a total of 470 MB of online storage. The other nodes were all AT computers acting as independent nodes that answer calls without sharing data contained on their fixed disks. All of the systems are connected by the IBM Token-Ring Network. Three of the nodes were using the Hayes™ 2400 Smartmodem™ and the rest were using the Hayes 9600 V.42 Series (error-correcting) modems.

At the time of this writing, the PCUG BBS has been expanded to 12 nodes. The demand for information through PCUG BBS has necessitated this expansion in order to provide availability at any time to the worldwide callers seeking information. Additional expansion is planned through the end of this

year. So, access to your product information by this worldwide user audience is only a phone call away!

The catalog information is also available electronically through the IBM Customer Support System (CSS). CSS is accessible to all Authorized IBM Personal Computer Dealers, Industry Remarketers-PC, and Special Product Dealers.

Business Partner Demonstrations: At Spring/COMDEX and Fall/COMDEX '88, IBM included several demonstrations of selected IOV adapters for Micro Channel computers in the IBM booth. This program was expanded for Spring/COMDEX '89 to include demonstrations of selected IOV bus master adapters for Micro Channel computers and included the participation with IBM of these business partners in the IBM booth.

Fall/COMDEX '89 Plan: It is IBM's intention to further expand their business partner program at Fall/COMDEX '89. A fourth edition of the *Catalog of Micro Channel Hardware Products* will be made available. The "bus master wall" will be expanded to include all currently available IOV bus master adapters. And, once again, selected IOV business partners will be asked to participate with IBM in demonstrating their bus master adapters for Micro Channel computers in the IBM booth.

Micro Channel Card Compatibility Certification

National Software Testing Laboratories, Inc. (NSTL) has independently developed procedures to test hardware adapters that attach to the Micro Channel bus for compatibility with currently available systems,

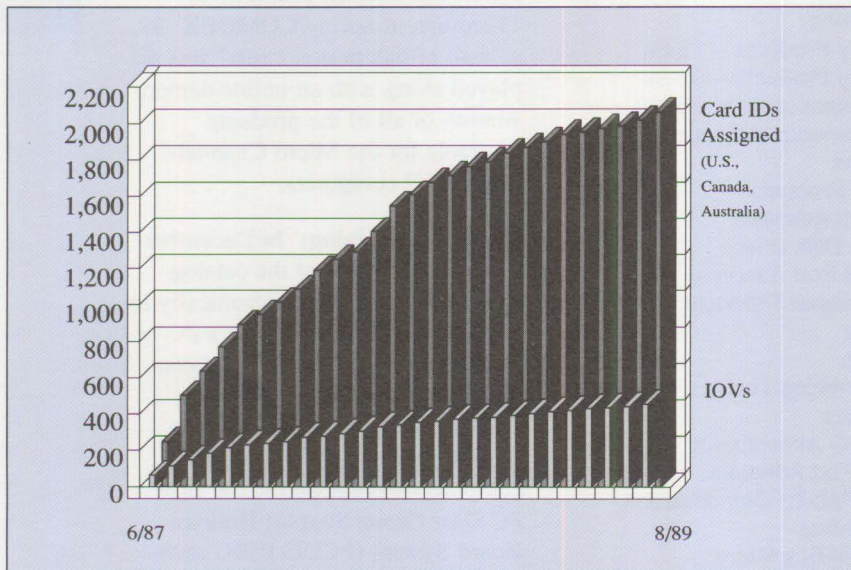


Figure 1. IOV Micro Channel Card ID Assignment Activity

adapters and end-user software applications.

NSTL is now offering this testing and certification program to developers of Micro Channel adapters. To obtain additional information about this program, you may call NSTL at:

1-215-941-9600.

All enrolled adapters for the Micro Channel bus included in the *Catalog of Micro Channel Hardware Products* that have been tested by NSTL and have received the NSTL Certification will be highlighted as being "NSTL Certified" when this information becomes available.

As NSTL Certification data is received, it will be included in the monthly updates to the IOV adapter information for the Micro Channel bus. These updates are made available electronically through the IBM PC User Group (PCUG) Bulletin Board System (BBS). The phone number for the PCUG BBS and access information for the Micro Channel Hardware Products are documented earlier in this article.

IOV Development Activity

Since June 1987 a total of 2,064 Micro Channel Card IDs have been assigned to 455 IOVs (see Figure 1). This figure represents only those IDs assigned through the IBM National Support Center to IOVs in the United States, Canada and Australia. Additional assignment centers are located in the United Kingdom for assignment throughout England and Europe, and in Japan for assignment throughout Asia.

Based on current IOV product enrollment information for the catalog received in the United States, there are 959 IOV products available for the PS/2 computers with the Micro

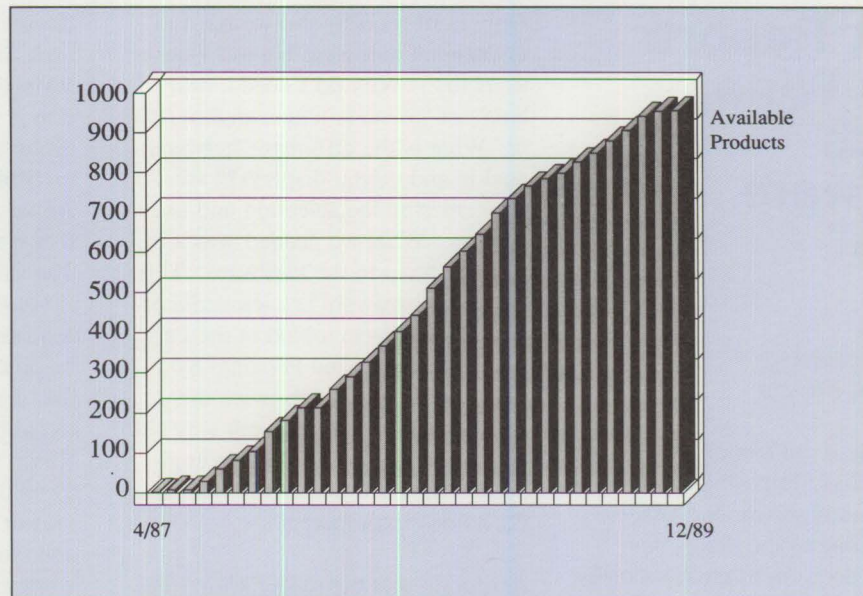


Figure 2. IOV PS/2 Micro Channel Product Availability

Channel bus from 305 IOVs (see Figure 2). These products represent IOVs worldwide, as the catalog distributed worldwide since 1988 includes an enrollment request that many developers in Europe and Asia have exercised to ensure that their products receive this merchandising advantage.

As of 9/8/89, the 959 PS/2 Micro Channel hardware products available from these 305 IOVs are as follows:

32%	Communication Adapters
23%	Storage Products
11%	I/O Adapters
7%	Data Acquisition Products
6%	Memory Products
3%	Displays, Display Adapters
3%	Printers and Scanners
2%	Debug Tools
13%	Other Products

This information represents just a snapshot in time, as new IOV products for PS/2 computers with the Micro Channel bus become available daily.

ABOUT THE AUTHOR

Mili Sanderson joined IBM in 1974. She has held a variety of marketing support positions in Office Products, Data Processing, National Accounts and Entry Systems Divisions. She became a member of the PS/2 development planning group in 1983, and since January 1988, has been part of the special marketing team assisting with IOV development for the Micro Channel. She is the primary outbound interface to the IOV community, offering participation in IBM's Micro Channel publicizing efforts to vendors.

Book Report: "The Winn Rosch Hardware Bible"

Chet Heath
IBM Corporation
Boca Raton, Florida

The last time I did a book report, it was for a Miss Florence Kennedy as a senior in high school. That experience is one of the things that shaped my decision to pursue a technical career. Now, 25 years later, my technical degree has failed me and I have another book to review. This time the effort is voluntary. Indeed, I am grateful that the book exists because it solves a real problem for me and many others (which is more than I can say for *Beowulf* and Miss Kennedy).

The problem that many people now need to solve is this: How do we understand the hardware side of personal computer systems? Most users have approached the PC or PS/2 computer as a tool to execute a piece or group of software. Like our car, VCR, or dishwasher, we accept that it works and don't want to ask why. When the PC was announced, and with each addition and extension, the bookstores filled with a selection of guides to software selection, operation or creation. Peter Norton went so far as to define the software designer's interface to the hardware. No one explained, for the non-engineer, how the hardware systems worked.

Only until recently did we need to understand hardware issues. Questions like, "Why do I need a new bus?" or "What is a bus, anyhow?" or "What is the difference between analog and digital displays?" are now entering the selection and use process. What we needed was a PC jockey's map to the hardware. This book is a hardware encyclopedia of all the systems involved in the Personal Computer and Personal System/2. It's well worth twice the purchase price. If I had my way I'd give a copy to every systems engineer in the field who deals with the PS/2 line of computers.

Assuming that the reader knows little more than which end of a soldering iron to hold, the author explains microprocessors to memory maps, acoustic couplers, zero wait state operation, and everything in between. The index works and one can use this book for reference or read it cover to cover (but not in one sitting). Opening the book at random, the page falls to "TMS34010," a video processor used by a competitor to build an advanced display card. *Note:* This book isn't all IBM (that's good), nor is it biased in any direction except toward fact and understanding. Another section explains serial communication and compatibility to Intel chipsets, still another floppy and hard disk operation and controllers. The list of topics is a book in itself, almost everything is there and it all makes sense, although you often get more by rereading some sections.

Is it flawless? No, but we'll forgive much in a first book of this type. The description of EISA is many months old and based on the public

domain descriptions available at the time; it might have been better to leave this topic out until a specification is publicly available and can be discussed. I wish there were more on Motorola-based systems, like Apple and Commodore, but perhaps that is volume 2, and is yet to come. The title needs improvement: *The Winn Rosch Hardware Bible*. That sequence of words is not likely to be picked up on any keyword scan that deals with PC or one of the internal topics of interest. Winn Rosch is well known in the industry as an unbiased and knowledgeable reviewer of adapter designs, systems and, yes, bus architectures. His work appears in most publications that respond to the keywords "Personal" or "Computer." Once you are aware of that connection, you know that this book is useful.

So how do you get one so you can find out what's going on in your personal system? First of all, if your bookstore doesn't have it, they probably can get it for you. Give them the title, author, ISBN number (ISBN = 0-13-160979-3) and the publisher (Brady, division of Simon and Schuster). Easier still, there is an 800 number to call: 800-624-0023 in the U.S. European readers can call 201-767-5937 (in New Jersey). In the U.K. call 442-231-555.

New Products

Hardware

New Models of IBM Personal System/2 Model 70 386 (8570-061 and 8570-A61)

The PS/2 Model 70 family is expanded with two new models, the 8570-061 and the 8570-A61. The 8570-061 provides 20 MHz 386 performance with 2 MB of high speed memory and a 60 MB fixed disk. The 8570-A61 provides 25 MHz 386 performance with 2 MB of high speed memory, 64 KB memory cache, and a 60 MB fixed disk. These models with 60 MB fixed disk capacity allow the Model 70 386 family to address a wider range of customer needs.

Highlights:

- 80386 32-bit microprocessor
- System board memory expandable to 6 MB (8570-061) or 8 MB (8570-A61)
- 60 MB fixed disk with integrated controller
- Micro Channel architecture with one 16-bit and two 32-bit slots
- Optional 80387 math co-processor
- Optional 486/25 Power Platform (8570-A61)
- Ergonomic desktop design

PS/2 Model 30 286 with 30MB Fixed File

The PS/2 Model 30 286 (8530-E31) is a 30 MB fixed disk drive version of the Model 30 286 and has 1 MB memory standard on the planar. The Model 30-E31 utilizes the Intel

80286 processor operating at 10 MHz with one wait state to system memory and has the following integrated functions:

- Parallel port
- Serial port
- Pointing device port
- Keyboard port
- 1.44 MB diskette drive support
- VGA graphics capability

Highlights:

- 30 MB fixed disk capacity
- 1 MB memory standard on the system board
- Enhanced price/performance ratio
- Memory can be increased to 2 MB or 4 MB on the system board
- IBM enhanced keyboard

IBM PS/2 Model 30/70 Base Memory Enhancement

The following models of the PS/2 product family have been enhanced by increasing the standard memory at no additional charge. The following table illustrates the new configurations.

Description	Standard Memory	
	Original	Enhanced
8530-E01	512 KB	1 MB
8530-E21	512 KB	1 MB
8570-E61	1 MB	2 MB

IBM PS/2 Enhanced 80386 Memory Options

IBM announces four megabit memory technology for the IBM Personal System/2. The PS/2 Enhanced 80386 Memory Options are the first IBM products to use four megabit memory technology.

The announcement of four megabit memory technology demonstrates our commitment to provide the PS/2 user with the latest technology available to support Operating System/2 and OfficeVision as primary participants in Systems Applications Architecture, and AIX PS/2 as a key participant in the AIX Family definition. The IBM PS/2 Enhanced 80386 Memory Options provide the IBM PS/2 Models 70, P70, and 80 with the ability to expand system memory up to 16 megabytes with the installation of a single adapter using a single expansion slot.

Highlights:

- First IBM product using IBM four megabit memory technology
- One adapter provides PS/2 Models 70, P70, and 80 with 16 MB capacity using a single expansion slot

IBM PS/2 Expanded Memory Options

The All Chargecard™ provides the IBM Personal System/2 Model 30 286 with an advanced memory management unit. The All Chargecard supports the Lotus® / Intel / Microsoft® Expanded Memory Specification 4.0 (LIM EMS 4.0), the IBM Workstation Connectivity Memory Manager Enhancement, and Software Carousel™ Version 3. The All Chargecard can be used to load device drivers and terminate and stay resident (TSR) programs into unused memory addresses between 640 KB and 1024 KB. This capability can be used in some LAN and host-attach environments to increase the amount of memory available for other applications. The All Chargecard installs directly on the system board and does not require an open option card slot. The

All Chargecard works with all IBM memory installed on the system board or in an IBM memory card installed in an option card slot.

The 3.0 MB Expanded Memory Adapter combines six IBM 0.5 MB Memory Module Kits with the IBM PS/2 Multifunction Adapter and the All Chargecard. Installation of this adapter on the system board and in an option card slot of the PS/2 Model 30 286 provides 3.5 MB of expanded memory when combined with the 0.5 MB of memory standard on the system board.

The 4.0 MB Expanded Memory Option combines two IBM 2.0 MB Memory Module Kits with the All Chargecard. Installation of this option on the system board of the PS/2 Model 30 286 provides 4.0 MB of expanded memory without the use of an option card slot.

Highlights:

- Sixteen sets of 256 registers for full LIM EMS 4.0 compatibility
- Provides up to an additional 288 KB for DOS applications running in some LAN and host-attach environments
- Supports IBM Workstation Connectivity Memory Manager Enhancement
- Supports Software Carousel Version 3
- Enhanced LIM EMS 4.0 Support for the PS/2 Model 30 286

IBM PS/2 Model 25 720 KB 1-Inch High Diskette Drive Option

The PS/2 720 KB 1-Inch High Diskette Drive Option (#1056) allows the user to install a second 3.5-inch

720 KB diskette drive in a PS/2 Model 25 (8525-001, G01, 004, and G04). This diskette drive is functionally equivalent to feature #4106. Feature #1056 must be installed in Model 25s with serial numbers greater than 100,000. (For visual reference, the first diskette drive indicator light must be below the media insertion slot.) For installed Model 25s with serial numbers less than 100,000, continue to order feature #4106.

Highlights:

- 3.5 Form Factor
- 1-inch high diskette drive
- Standard function (read/write/format)
- Access time: 6 ms
- Track density: 135 TPI
- Transfer rate: 250 K bits per second

Audio Capture and Playback Adapters and Video Capture Adapter/A for the IBM PS/2

The Audio Capture and Playback Adapters and Video Capture Adapter/A are PS/2 adapters providing the capability to capture, digitize, and playback high quality audio and images on standard PS/2 hardware. The cards are easy to use and work with standard audio and video input devices, including video cameras, video cassette players, microphones, compact disc players, and cassette players. The Audio Capture and Playback Adapters also support a variety of audio output devices for playback, including headphones, speakers, and amplifiers. The adapters work in conjunction with the Audio Visual Connection

program. AVC is an application enabling program that allows users to create and present high quality, interactive, multi-media presentations.

The PS/2 Audio Capture/Playback Adapters and PS/2 Video Capture Adapter/A are manufactured by Rexon/Tecmar, Inc. exclusively for resale and service by IBM.

Highlights:

- Audio Capture and Playback Adapter
 - Enables users to capture/playback near-CD quality audio
 - Easy-to-use audio capture
 - Audio can be manipulated and used in high quality presentations created with the IBM Audio Visual Connection
 - Uses standard audio-input devices (microphone and line level)
 - Uses standard audio-output devices (speakers, headphones, or amplifier)
 - Uses standard mini-stereo and mini-phone jacks
 - Hardware compression and decompression
 - Programmable digital signal processor (DSP) for independent application development
- Video Capture Adapter/A
 - Enables user to capture high quality, photo-like images
 - Full frame capture from motion video in 1/30 second
 - Uses standard video input/output devices (RGB, NTSC and Y/C)

- Captures 65,536 colors with a resolution of 640 x 480 x 16 bits
- Display capability allows user to view live video during capture process; the card is not required for general image presentation purposes
- 614 KB dual-ported Video RAM buffer
- Various image display modes (live, memory, overlay, transparent, pixel)

IBM 8209 LAN Bridge

The IBM 8209 LAN Bridge interconnects an IBM Token-Ring Network and an Ethernet Version 2 or IEEE 802.3 Local Area Network. Systems and workstations with compatible protocols such as TCP/IP, OSI, SNA, NETBIOS, or IEEE 802.2 can communicate across this connection. The 8209 handles all necessary conversion to route information between the dissimilar LANs. Token-Ring stations view the 8209 as a bridge to another Token-Ring. To Ethernet / IEEE 802.3 stations, the 8209 LAN Bridge is functionally transparent.

Highlights:

- Bridges 4 or 16 Mbps IBM Token-Ring Network to Ethernet Version 2 or IEEE 802.3 LANs
- Supports Ethernet Version 2 or IEEE 802.3 simultaneously
- Bridge configurations allow multiple protocols (TCP/IP, OSI, SNA, NETBIOS IEEE 802.2)
- Compatible with IBM LAN Manager program
- Customer setup (CSU) suitable for table-top or rack shelf mount

IBM Token-Ring Network Remote Program Load

The IBM Token-Ring Network Remote Program Load feature allows IBM Personal Computers or Personal System/2 Models 25 and 30 computers, with the appropriate Token-Ring Adapter, to gain access to the Token-Ring network and request program loads, even though the device may not have a disk or diskette drive. The feature is required for each of the devices on the network that have a need to be loaded from a controlling station. The Remote Program Load (RPL) (#7839) is a module designed to be plugged into a designated socket on the IBM Token-Ring Network PC Adapter II (#9858).

IBM Personal Typing System/2, IBM Personal Typing System/2-286, and IBM Personal Typing Solution Upgrade

The Personal Typing System/2 and the Personal Typing System/2-286 meet the need for an integrated secretarial workstation with DOS or OS/2 capability. The Personal Typing System/2 system unit utilizes an 8086 processor, has two full size PC/XT card slots, 640 KB RAM, and comes standard with one 1.44 MB flex disk drive and system unit keylock.

The Personal Typing System/2-286 utilizes an 80286 processor, has two full size AT card slots, 1 MB RAM, memory expansion to 4 MB on the system board, and comes standard with one 1.44 MB flex disk drive and system unit keylock.

Models for both the Personal Typing System/2 and Personal Typing System/2-286 include a mono-

chrome or color display, Space Saving or Enhanced PC Keyboard, and the Personal Typing System Version 3.0 program. Connectivity is supported on the Personal Typing System/2 and Personal Typing System/2-286, including System/370, System/36, System/38, AS/400, 5520, baseband, broadband, and Token-Ring Local Area Networking.

The Personal Typing Solution Upgrade includes the Version 3.0 program and either the Correcting Wheelwriter® or Correcting Quietwriter® printer for attachment the PS/2 system units. The Personal Typing Solution Upgrade will only be available through authorized typewriter dealers certified to sell the Personal Typing System.

IBM Personal Page Printer II Model 031

The IBM Personal Page Printer II Model 031 is a new model in the IBM PostScript® laser printer family. It incorporates all the features and function of the IBM Personal Page Printer II (4216 Model 030) plus enhanced function including:

- Support of a dual bin and envelope sheet feeder feature
- Software emulation mode switching capability
- Diablo® 630 emulation
- Addition of 4 Helvetica® Narrow typeface styles, bringing the total standard resident typeface styles to 47.

The new Personal Page Printer II Model 031 (4216-031) will replace the existing IBM Personal Page Printer II (4216-030). It exemplifies IBM's commitment to continue offering function- and price-competitive PostScript laser printers. A model upgrade kit will be available

for customers with an IBM Personal Page Printer II (Model 030) to bring their printer up to functional equivalency with the IBM Personal Page Printer II Model 031.

Highlights:

- Compact, tabletop laser page printer
- Up to six pages/minute for letter-size paper
- 300 x 300 dots per inch (dpi) resolution
- Very quiet (52 dBAI printing, 48 dBAI idling)
- Optional dual bin and envelope sheet feeder
- Variety of emulations: Proprinter XL, Hewlett-Packard LaserJet Plus® and Diablo 630
- Two output trays for sequenced or unsequenced output
- Various paper sizes and weights (including legal-size)
- Envelopes, transparencies, and labels
- Initial set of supplies included (toner cartridge and photoconductor unit)
- Built-in PostScript interpreter with 47 typeface styles
- Automatic emulation mode switching from the workstation
- 16.67 MHz M68000® processor
- 2 MB of random access memory, field-upgradable to 4 MB
- Flexible connectivity: serial, PC Parallel, AppleTalk®

IBM PS/2 300/1200/2400 Internal Modem /A Supported on Additional Models

The PS/2 300/1200/2400 Internal Modem/A high-function internal Modem now supports PS/2 Models 50, 55 SX, 60, and 70. This modem was previously announced to support the Model P70. The PS/2 300/1200/2400 Internal Modem/A is Hayes Compatible* and provides standards compatibility via CCITT V.22 bis, Bell 212A, and Bell 103 full duplex operation. Most functions of the new modem are also compatible with the IBM 5853 External Modem. Functions include automatic dial, automatic answer, and the ability to direct-connect into US public or private switched network telephone lines. A half-length, 16-bit PS/2 Micro Channel adapter, the modem includes NS16550A FIFO asynchronous support, eight programmable port assignments, and system speaker for telephone line monitoring.

* Certain unique functions are not supported. See Technical Reference Manual for details.

Highlights:

- PS/2 Micro Channel adapter versatility
- Compatibility with PS/2 Dual Async Adapter/A
- NS16550A asynchronous FIFO communications controller
- Programmable port assignment (COM1 to COM8) via POS
- Interrupt control for level-sensitive interrupt levels 3 and 4

- System speaker for telephone line monitoring
- Most prevalent standards at each speed:
 - 300 bps: Bell 103
 - 1200 bps: Bell 212A
 - 2400 bps: CCITT V.22 bis
- Full duplex operation
- Call progress signal (dial tone, busy tone, answer tone) detection
- Direct telephone line connection via standard modular connector to public or switched lines
- Self-diagnostics including analog/digital local/remote loop back test

System/370 Channel Emulator/A for IBM PS/2 Models 50, 55, 60, 70, P70, and 80

The IBM System/370 Channel Emulator/A is now available for the PS/2 Models 50, 55, 60, 70, P70, and 80. This card, which emulates the operation of an IBM System/370 channel, coupled with the Remote PrintManager software, may be used to channel attach IBM 3820, 3825, 3827, or 3835 Page Printers to the IBM PS/2 Models 50, 55, 60, 70, P70, and 80. The new S/370 Channel Emulator Technical Reference provides information about the programming interface which is needed to develop appropriate software to channel attach other System/370 devices.

Highlights:

- Support warranted for connection to the IBM 3820, 3825, 3827, and 3835 Page Printers

- Single card adapter for PS/2 Models 50-80
- Options diskette supports installation, test, and PCFRIEND demonstration program
- Byte multiplexer, block multiplexer, or selector channel protocol may be emulated
- Interlocked and data streaming (3.0 and 4.5 Mbps) protocols may be emulated
- Hardware support for initial selection and data transfer System/370 channel sequences
- On-card 64 KB high speed memory, providing read/write record size up to 64 KB
- Technical reference publication provides the programming interface

Software

OS/2 Standard Edition 1.2 and OS/2 Programming Tools and Information Version 1.2, Available with Additional Functions

IBM OS/2 Standard Edition Version 1.2 and IBM OS/2 Programming Tools and Information Version 1.2 are now available with new functions.

The OS/2 Information Presentation Facility provides easy-to-use functions for the application developer to define help panels and to service end-user help requests. Traditional hardcopy Command Reference information has been placed online to increase usability. A new Dual Boot Utility allows OS/2 and IBM DOS to boot from the same fixed disk drive. Further, enhancements will

be made in establishing file system file names and a new printer device driver will be added in UPDATES to the OS/2 Standard Edition Version 1.2 products. These UPDATES will be available November 30, 1989.

All previously announced function is also available today, including a new Desktop Manager interface featuring iconic representation and a new File Manager that permits direct manipulation. A Systems Applications Architecture Dialog Manager, a High Performance File System, additional device support, and a windowed System Editor are also included. Compatibility with DOS V4.0 at the API level, and significant reliability, availability, and serviceability enhancements are also provided.

To assist OS/2 application developers, the OS/2 Programmer's Toolkit V1.1 and the OS/2 Technical Reference V1.1 have been combined into a single product, IBM Programming Tools and Information Version 1.2. This new product has been enhanced to support the new features and functions provided in IBM OS/2 Standard Edition Version 1.2.

Highlights:

- Increased usability through online Command Reference
- OS/2 Information Presentation Facility
- Dual Boot of OS/2 and DOS from same fixed disk drive
- New File, Desktop, and Print Manager interfaces
- Availability of the SAA Dialog Manager
- High performance file system

- New printer device drivers supporting IBM and other manufacturers' printers
- OS/2 programmers Toolkit V1.1 and OS/2 Technical Reference V1.1 combined into a single product, IBM OS/2 Programming Tools and Information V1.2, which includes support for Dialog Manager and OS/2 Presentation Facility

IBM PS/2 RPG II Application Platform and IBM PS/2 RPG II Application Toolkit Version 1.1

Version 1.1 of the PS/2 RPG II Application Platform and Toolkit includes all the support found in Version 1.0, plus: shared resource/LAN connectivity; up to 16 workstations (up to 8 of which may be async-connected); spooler enhancements; messaging between workstations; improved operator controls; a backup/restore function; and a Development Support Utility.

Highlights:

- Automatic updating of alternative index files
- Support for OUTPUT/ADD to a sequential file from multiple workstations simultaneously

CICS OS/2 Version 1.11 Available With Functional Enhancements

CICS OS/2 Version 1.11 is an update to CICS OS/2 Version 1.1. CICS OS/2 Version 1.11 provides functional enhancements within the OS/2 Extended Edition operating system environment. These enhancements are improved systems operation through the automatic

shipment of CICS OS/2 terminal definitions to a host CICS system during transaction routing, and support for double byte character sets.

Highlights:

- CICS OS/2 shippable terminal support
- Double-byte character support
- Consolidation of service
- Source line debug

IBM AIX PS/2 Operating System Version 1.1 – Additional Support

The AIX PS/2 Operating System Version 1.1 supports additional IBM personal computers, displays, printers, and features. New systems support includes IBM PS/2 Model P70 386 and the IBM Industrial Computer.

ORACLE® RDBMS

Version 6 for IBM AIX PS/2

ORACLE for the IBM AIX PS/2 Operating System is a distributed relational database management system (RDBMS). ORACLE offers relational database and Structured Query Language (SQL) capabilities, multi-user support, an active data dictionary, and security facilities. The transaction processing option is a built-in feature of the RDBMS and provides improved transaction performance to the database. The transaction processing option includes a new concurrency control mechanism, the Row Lock Manager. Features of the Row Lock Manager include row level locking and row level multi-versioning. SQL*DBA, included in the RDBMS, is an interactive utility for database administration and performance analysis that can be used to

manage local and remote nodes. The AIX PS/2 Operating System Version 1.1, 1-16 user option is required to support more than 2 ORACLE users.

Other available offerings for AIX PS/2 Operating System include SQL*Plus®, an interactive user interface, SQL*Forms™, SQL*Menu™, and SQL*Net TCP/IP. SQL*Net TCP/IP (Transmission Control Protocol / Internet Protocol) provides database networking facilities and distributed database capabilities. In addition, programming interfaces for C and FORTRAN languages are provided.

Highlights:

- Higher transaction rates
- Faster response times
- Increased fault-tolerance
- Increased support for simultaneous online transaction processing, decision support, and reporting
- Increased support for very large databases

PC Node Manager Version 1.1

PC Node Manager (PCNM) Version 1.1 is an enhanced and renamed version of PC Node Executive, which it replaces. It is a licensed program offering for network based PS/2, PC XT, AT, or PS/55 that require data and resource distribution and administration control from an IBM 3090™, 308X, 43XX, or 9370 processor. PCNM Version 1.1 is operational under OS/2 Extended Edition or DOS. It uses the 3270 emulation functions that are available for these systems.

PCNM provides the ability for an IBM 3090, 308X, 43XX, or 9370 host processor product, NetView Distribution Manager, to distribute and retrieve software and data to and from workstations directly or indirectly via operation with the LAN server files. PCNM Version 1.1 requires the use of the corequisite product, VTAM Protocol Conversion Application Release 2, or VTAM Protocol Conversion Application Release 1 to access NetView Distribution Manager. PC Node Manager Version 1.1 and VTAM Protocol Conversion Application Release 2 can all have transmissions between each other in compressed and compacted form. This reduces line utilization and transmission time.

Highlights:

- IBM PC, PS/2, and PS/55 computers that are directly attached to MVS/ESA™, MVS/XA™, MVS/SPT™, VM/XA™, VM/SP, or VM/SP HPO can use NetView Distribution Manager facilities
- "In-line" function to compress/decompress and compact/decompact is provided for improved data transmission performance to or from the host NetView Distribution Manager
- OS/2 Extended Edition Version 1.1, Version 1.2, and OS/2 Extended Edition (J) are supported by PCNM
- The Japanese IBM PS/55 is supported under OS/2 Extended Edition Version 1.1, Version 1.2, and the OS/2 Extended Edition (Japanese) Version 1.1. The support is for English language mode only
- PCNM is operational with the IBM Personal Communica-

tion/3270 product that runs in DOS

- Users can do a PCNM start (which does a load and run) and stop (which also does a PCNM unload), without operator involvement
- PCNM nodes can be attached to a MVS or VM processor via a DOS 3270 emulator or the OS/2 EE Communications Manager's 3270 emulator.

NetView/PC™ Version 1.2.1

NetView/PC is an extension to the Communication Network Management (CNM) services provided by NetView. NetView/PC Version 1.2.1 updates the NetView/PC Version 1.2 by adding the Remote Console Facility (RCF) as well as adding Application Program Interface / Communication Services (API/CS) access to asynchronous communications through the OS/2 Extended Edition Communications Manager.

IBM Classroom LAN Administration System Version 1.20

IBM Classroom LAN Administration System Version 1.20 is an educational product that combines software and documentation to help manage courseware in a LAN environment. It simplifies network installation, helps a teacher control student access to courseware, and reports each student's use of courseware. It reports performance data for IBM courseware that tracks student progress. The product is appropriate in school environments requiring a local area network.

Highlights:

- Number of user IDs increased from 1000 to 2500
- Ability to make a printer on a workstation function as a print station for the other workstations on the network (must have Netware 2.12 or later)
- Ability to backup system to tape
- Increased the number of courseware programs that can be installed from 200 to 400
- Application programs have access to the list of students in a class
- Advancement of students to the next grade level at the end of the school year
- Improved security features that limit disk space and prevent concurrent logons via the LAN (must have Netware 2.12 or later)
- Provides new Application Program Interface for developers to identify "partners" that run application programs under IBM Classroom LAN Administration System 1.20
- Cleanup utility for removal of unnecessary files
- Permanent assignment of printers to workstations

Audio Visual Connection™ (AVC)™ Version 1.0

The Audio Visual Connection (AVC) licensed program is a PS/2 application enabling program providing authoring software, file management capability, and a run time facility. Applications can communicate with users through hypertext linking techniques and digitized, high quality audio (between AM

and FM level) and photo-like color images. A multitude of sound, display, and motion effects are provided to make this interchange eye-catching, fast paced, and attention holding. The AVC can be used to develop stand-alone applications or as a presentation service for other programs. Highly interactive cooperative processing is allowed through the AVC interface with the host expert system program, KnowledgeTool 2.1.1. The AVC, in conjunction with the Audio Capture and Playback Adapters and Video Capture Adapter/A, provides a total audio-visual solution for the application developer. The AVC runs under OS/2 EE 1.1 or 1.2, OS/2 SE 1.1 or 1.2, or DOS 4.0 with expanded memory support.

Highlights:

- Desktop audio-visual capability using high quality, photo-like color images and high quality (between AM and FM) audio
- Access to many types of images, drawings, and text screens using Video Capture Adapter/A, scanned image, image conversion and copy screen programs
- High quality presentation text – anti-aliased, kerned, proportionally spaced, in multiple styles and sizes
- Audio options – voice quality, music quality, stereo music quality
- Highly interactive applications allowed through the use of the Audio-Visual Authoring language
- Hypertext-like link capability
- HyperHelp – a hypertext-like online help for animated demos of product functions

- Run time component distributed at no additional charge
- PS/2 audiovisual presentation services front-end for IBM host Expert System product, KnowledgeTool™
- All editors and capture facilities designed for ease of use
- Flexible user interface – keyboard or mouse

IBM Token-Ring Network Bridge Program Version 2.1

IBM Token-Ring Network Bridge Program Version 2.1 provides connectivity between two local or remote rings of a 4 Mbps and/or 16 Mbps IBM Token-Ring network. In the remote ring environment, the rings are connected via a leased teleprocessing (TP) line. IBM Token-Ring Network Bridge Program Version 2.1 communicates with IBM LAN Manager Version 2.0 to provide network management capability for the multi-ring environment.

Highlights:

- Frame filtering capability for local bridge
- 7820 Integrated Services Digital Network (ISDN) Terminal Adapter attachment

IBM Windows Connection

IBM Windows Connection is a licensed program that enables 3270 emulation programs that communicate via HLLAPI commands to be run under Microsoft Windows. Specific support is provided for IBM Personal Communications/3270 and PC 3270 Emulation Program Entry Level Version 1.2.1. This Windows application supports multiple 3270 sessions, file transfer to and from the host, and contains features that

enable the user to personalize their 3270 sessions. IBM Windows Connection provides seamless integration for end users with Microsoft Windows and 370 host application requirements.

Highlights:

- Communicates with the HLLAPI interface for access to multiple, independent 3270 sessions
- Supports file transfer from the 3270 session to and from a DOS/Windows session
- Enables keyboard customization via the KeyMapper facility
- Provides facilities that allow the user to personalize the 3270 session display colors and fonts
- Supports CUT and DFT-mode (including IBM LAN) terminal emulation
- Provides support for multiple screen sizes (3278 terminal Models 2 through 5).

IBM Storyboard™ Plus Version 2.00

IBM Storyboard Plus Version 2.00 is a "business multimedia" application that combines drawing, painting, text, still video, music, and voice to produce high quality, animated, on-screen audiovisual presentations in an IBM PC, certified IBM-compatible, or IBM PS/2 computer. Upgrades from IBM PC Storyboard Version 1.00, 1.11 and from IBM Storyboard Plus Version 1.00, 1.01 to Storyboard Plus Version 2.00 are available.

Highlights:

- New functions to enhance picture and story editing capabilities

- Improved performance and usability over Storyboard Plus Version 1.01
- Improved documentation over Storyboard Plus Version 1.01
- Support for a number of video capture interface cards
- Support for additional printers, including additional color printers
- Support for the IBM 8514/A Video Adapter in 640 x 480 – 256 color mode
- Ability to mix display resolutions within a presentation
- Ability to easily combine and transfer a story to diskette

Workstation Interactive Test Tool V1.0

The IBM Workstation Interactive Test Tool provides a regression testing function for interactive applications in the following Systems Application Architecture (SAA) environments:

- MVS/XA and MVS/ESA (TSO-E, CICS/MVS, and IMS/TM)
- VM/SP and VM/XA
- OS/400™
- OS/2 Extended Edition

Workstation Interactive Test Tool provides facilities to automatically record and re-execute interactive application test sessions. An SAA Common User Access (CUA) interface enables the query and display of test results and the analysis of test case discrepancies.

Highlights:

- Automated testing

- Interactive and partial-screen comparisons
- Test case customization
- Batch execution
- Detail and summary reports
- Screen printing
- Teleprocessing Network Simulator test support

IBM Expert System Consultation Environment/PC Available

IBM Expert System Consultation Environment/PC (ESCE/PC) provides customers with a workstation tool for executing knowledge-based applications created on Expert System Environment/VM or Expert System Environment/MVS. It provides for deployment of knowledge-based applications under DOS on IBM Personal Computers, and allows users who are not experts to access expert knowledge as a decision-making resource.

The IBM PC AT (except AT/370) and the IBM PS/2 Models 30-286, 50, 60, 70, and 80 are supported.

Geo-Information Workstation Supervisor Version 1.1.0

Geo-Information Workstation Supervisor 1.1.0 enhances the function and usability of the current release for:

- Storage of application menus outboard at the PS/2
- Correlation between display cursor and digitizer pointing positions
- Application workspace size
- Attachment of 19" monitor

- Expanded plotting and printing capabilities
- Pop-up local function and help panels
- IBM mouse and OEM digitizer support.

Highlights:

- Outboard application menus
- Correlated cursor
- 4 K x 4 K application workspace at workstation
- IBM 7554 19-inch display support
- Local plot and print enhancements
- Host (ISPC) plot concurrency
- Save / replay graphic screen
- IBM mouse and OEM Digitizer support

Enterprise Management Control Series for AIX and PC/DOS

The Enterprise Management Control Series (EMCS) is a set of licensed programs that provides management with a series of tools, based on critical path method (CPM) of project scheduling, to help improve overall control of time, cost, and resources for industrial sector projects.

A key feature of the EMCS design enables information to be gathered and analyzed at any level of an organization to show the inter-relationships of project activities in a single or multiple project environment. The flow of this information can be passed from multiple CPU platforms in a distributed network to provide specific views of program or project status. Because of this design, managers can now allocate

and manage resources across multiple organizations and/or projects, thus helping to maximize the use of key resources.

OSL PLA for PS/2 AIX

Optimization Subroutine Library (OSL) for linear, quadratic, and mixed integer programming is a high-performance library of mathematical programming algorithms and functions providing alternative solution methods and performance options for use with application programs that solve optimization problems on the PS/2 Models 70 and 80. OSL extends the size and type of problem that may be solved today. It consists of subroutines that are callable from FORTRAN which take advantage of state-of-the-art techniques for providing optimum performance. OSL is modular and can be called by the user at different levels: high-level routines which solve a general problem or low-level routines which can be used to build a tailored algorithm for a particular type of problem. OSL on PS/2 is functionally compatible with OSL for MVS, VM, and AIX/370.

Highlights:

- High-performance linear programming techniques using primal and dual simplex algorithms and interior point algorithms
- High-performance quadratic programming techniques
- High-performance mixed integer programming techniques
- Support for calling from VS FORTRAN application programs
- Solutions for problems with the number of constraints limited only by system storage availability

- High-level routines to solve problems with general-purpose approaches in a minimum of calls
- Lower-level routines that allow the caller to tailor the solution strategy to the special characteristics of the problem
- Modularity that will allow users to insert their own versions of subroutines

IBM ExecPlan

IBM ExecPlan, a executive planning tool designed for IBM PCs, offers executives a means for managing project data. Each project contains its own activities, durations, resources, and costs. Users can structure, model, and analyze projects, and make decisions with regard to time, costs, and resources.

Highlights:

- Executive level project planning tool
- Easy-to-use menu driven program
- Converts cost matrix data to project detail information
- Preformatted reports and displays for project status
- Can exchange project data with Enterprise Management Control Series

IBM FAWN (Forms Available When Needed)

IBM FAWN, a new forms distribution and printing application, is an electronic "Forms File Drawer" that uses a host attached advanced function printer or an IBM Personal Computer attached to a high quality page or laser printer. Documents can be accessed and printed "on demand." The FAWN workstation on the PC can receive new or changed

documents from the host as soon as they are created or at a specified date and time. A user in a remote location or department can select the desired document from a menu screen and print the most current version on the local FAWN workstation.

Highlights:

- Forms available to users at specified time and date, controlled by the forms administrator
- Automatic update to remote workstations from the host – update interval determined at the workstation and may be changed to accommodate peak periods
- Groups forms by category for easy retrieval – user can establish groups of forms or documents which can then be printed as a group
- Search capability on form number or document name – eliminates scrolling from one form to another
- High-quality printing of forms and documents
- Host printing capability – advanced function printing data streams supported
- Online help functions
- Dial-up capability for remote workstation supported

IBM Plant Floor Series™ Distributed Automation Edition – Communications System/2 Release 1.1

The Communications System/2 Release 1.1 is a licensed program that provides a standard application program interface to plant floor devices and other communications facilities.

It allows the user to develop manufacturing or process control applications on IBM Industrial Computers, IBM Personal Computers, or a Personal System/2 using the Operating System/2 Extended Edition 1.1. This product will be available in March 1990.

Remote PrintManager Version 3.0

IBM Remote PrintManager (RPM) Version 3.0 provides new functions that allow a customer to connect selected channel-attached IBM Page Printers in remote environments. RPM Version 3 uses an IBM PS/2 equipped with a System/370 channel emulator card and a communications card. Support is now provided to:

- Manage a remote print spool from the print server workstation
- Download jobs from the host at night and place them on the spool for printing the next day
- Attach to PS/2 machines which use the Micro Channel bus
- Share the printer between the host or other workstations on a LAN
- Merge ASCII data with forms overlays downloaded from the host

Highlights:

- Remote attachment for channel-attached IBM 3820, 3825, 3827, and 3835 Page Printers
- Support for IBM PS/2 machines using the Micro Channel bus
- Remote print spool management from the print server workstation
- Local area network support
- LAN/Host printer sharing

Trademarks

IBM, OS/2, Personal Computer AT, AT, Personal System/2, PS/2, Quietwriter, RT, and Wheelwriter are registered trademarks of International Business Machines Corporation. AIX, Audio Visual Connection, AVC, KnowledgeTool, Micro Channel, MVS/ESA, MVS/XA, MVS/SP, NetView, NetView/PC, Operating System/2, OS/400, Personal Computer XT, PC XT, Plant Floor Series, Presentation Manager, Storyboard, Systems Application Architecture, SAA, System/360, VM/XA, and 3090 are trademarks of International Business Machines Corporation.

All Chargecard is a trademark of All Computers, Inc.
AppleTalk is a registered trademark of Apple Computer Incorporated.
AutoCAD is a registered trademark of Autodesk, Inc.
Channel/2 is a trademark of ComTech International, Inc.
CNT-MCA is a trademark of Core International.
COMDEX is a registered trademark of the Interface Group, Inc.
CORE International is a registered trademark of CORE International.
Diablo is a registered trademark of Xerox Corporation.
Ethernet is a trademark of Xerox Corporation.
FileSafe is a trademark of Mountain Computer, Inc.
GEM is a trademark of Digital Research, Inc.
GSS is a registered trademark of Graphic Software Systems, Inc.
Hayes is a trademark of Hayes Microcomputer Products, Inc.
Helvetica is a registered trademark of Linotype AG.
Intel is a registered trademark of Intel Corporation.
ISOLAN is a registered trademark of BICC Data Networks, Inc.
LaserJet Plus is a registered trademark of Hewlett-Packard Company.
Lotus is a registered trademark of Lotus Development Corporation.
METACOMP, Inc. is a trademark of METACOMP, Inc.
MicroMASTER is a trademark of AOX Incorporated.
Microsoft is a registered trademark of Microsoft Corporation.
Mountain is a registered trademark of Mountain Computer, Inc.
M68000 is a registered trademark of Motorola Corporation.
Netware is a registered trademark of Novell, Inc.
Novell is a registered trademark of Novell, Inc.
Oracle is a registered trademark of Oracle Corporation.
Personal Image is a trademark of ComTech Imaging Technology.
PostScript is a registered trademark of Adobe Systems, Incorporated.
Proteon is a registered trademark of Proteon, Inc.
ProNET is a registered trademark of Proteon, Inc.
PScomm2/4 is a trademark of METACOMP, Inc.
PSconnect is a trademark of METACOMP, Inc.
Smart Eight/2 is a trademark of HAAR Industries, Inc.
Software Carousel is a trademark of SoftLogic Solutions, Inc.
SQL*Plus is a registered trademark of Oracle Corporation.
SQL*Forms and SQL*Menu are trademarks of Oracle Corporation.
Ultra Clipper is a trademark of Pixelworks, Inc.
UNIX is a registered trademark of AT&T Bell Laboratories.
Video One is a trademark of Ariel Computer Corporation.
Windows is a trademark of Microsoft Corporation.
2400 Smartmodem is a trademark of Hayes Microcomputer Products, Inc.
i860, 386, 486, 80286, 80386, 80386SX, and 80486 are trademarks of Intel Corporation.

“Multiple devices can operate at the same time. (page 11)

“Micro Channel architecture was defined to support multitasking and multiprocessing. (page 44)

“Bus masters can greatly increase the processing power of systems. (page 52)

“Command chaining allows masters to decide which of several potential control paths to follow. (page 57)

“The SCB architecture supports many functions found in larger IBM systems designed to facilitate multiprocessing. (page 66)

“An IBM SCSI adapter is actually a ‘computer within a computer.’ (page 88)

“The PS/2 Wizard adapter uses the power of Micro Channel architecture to bring the i860 microprocessor to the PS/2 line of products. (page 95)

“Potentially, the effective data throughput between disk and computer can double. (page 96)

“The level-sensitive interrupts of the Micro Channel bus are well suited for interrupt sharing. (page 109)

6325-5004-00

