

The Realtime Interface Co-Processor Multiport/2 adapter

by S. C. G. Sykes

The Realtime Interface Co-Processor Multiport/2 is a programmable, multifunction adapter that extends the processing capabilities of the Personal System/2® and provides a solution to applications with unique communications requirements. Customized for speed and flexibility, the Multiport/2 is fully programmable and supports asynchronous, byte-synchronous, and bit-synchronous protocols on its eight communications ports. This powerful single-slot computer can handle functions that previously required processing by the PS/2®. Microcode on the Multiport/2 provides a real-time multitasking base on which custom applications can be built. This paper describes the Multiport/2, its microcode, system support software, and development tools.

The Realtime Interface Co-Processor Multiport/2 is a fully programmable single-slot computer within the PS/2® that features its own microprocessor, memory, and communication capability. Its potential uses include the following:

- Communication subsystem
- Synchronous gateway controller
- Cell controller for nonintelligent devices
- Multiterminal interface
- PS/2 coprocessor

Unlike other IBM communications adapters for the PS/2, the Multiport/2 has its own microprocessor that operates independently of the microprocessor of the host PS/2. This independent microprocessor permits interrupt handling, data conversion, protocol conversion, message packaging, data validation,

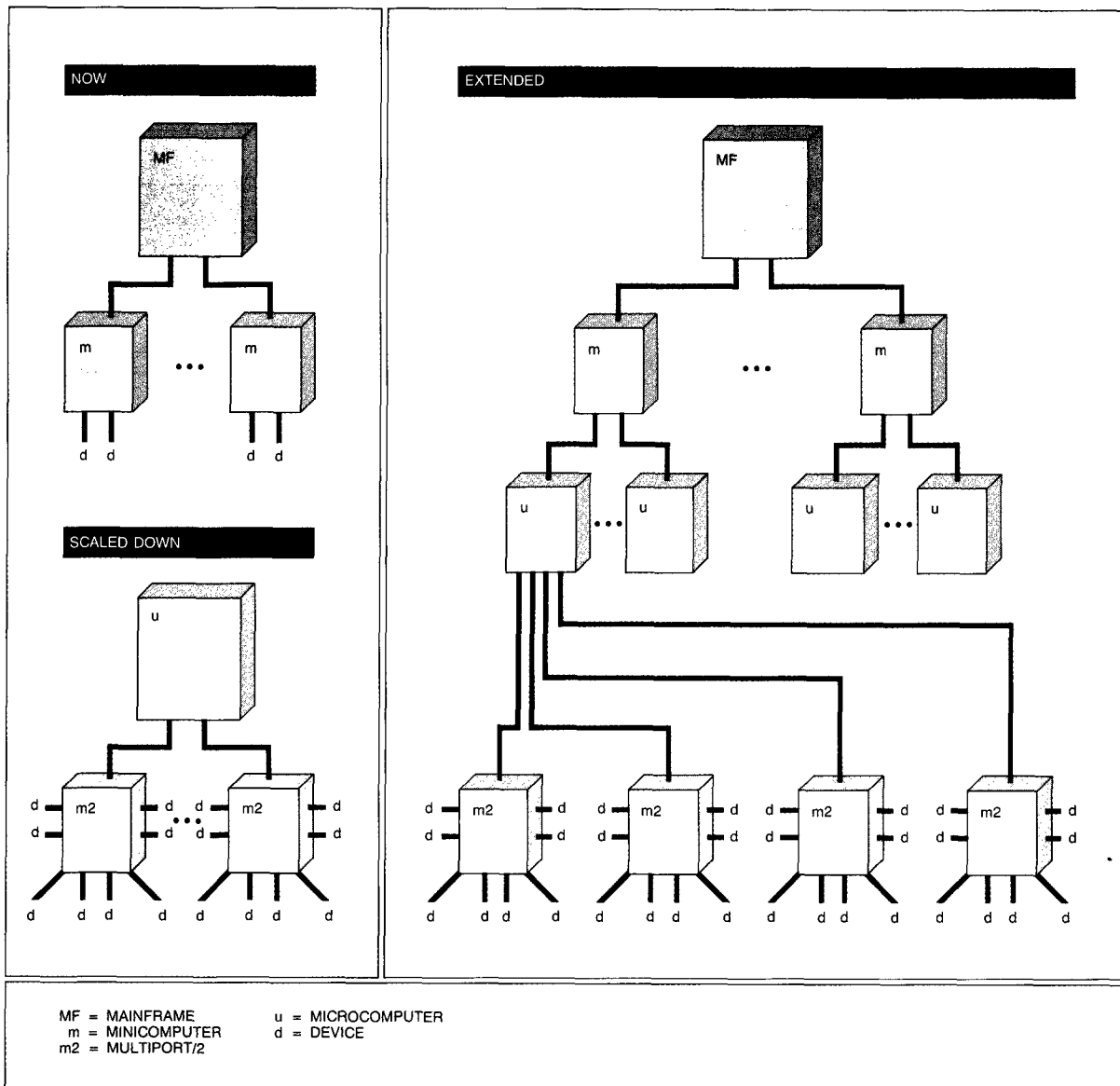
message acknowledgment, and message retransmission to be handled by the Multiport/2 without requiring processing by the PS/2. The Multiport/2 can be programmed to handle as much or as little of the communication processing as desired. As many as four Multiport/2s can operate simultaneously within one PS/2 to provide a multiprocessing environment.

Uses

The Multiport/2 can function as a communication subsystem for a PS/2 in much the same way that a minicomputer can function as a communication subsystem for a mainframe computer. Full-duplex communication with devices requiring different protocols and baud rates can be supported within one Multiport/2. An application using the Multiport/2 might have one communication port sending and receiving data using a byte-synchronous protocol to a mainframe computer, one communication port providing linkage to another PS/2, and several communication ports transmitting data using asynchronous protocols to low-speed serial printers. Each of the ports can be operating at different baud rates. While one port uses a standard byte-synchronous protocol, the other ports can communicate asyn-

© Copyright 1988 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

Figure 1 Systems scaled down and extended



chronously. The character size, parity, number of stop bits, and transmission rates of each asynchronous port are independent of the characteristics of any other port. Devices that use nonstandard protocols can be supported on the Multiport/2 by applications customized for those devices. The multi-protocol capability of the Multiport/2 allows it to replace multiple protocol-specific adapters when two or more protocols must be supported within one PS/2.

Devices requiring real-time monitoring or control can be supported from a PS/2 using the Multiport/2. In environments such as a plant floor, where alarms from nonintelligent devices may require real-time response, a Multiport/2 in a PS/2 may be used at a fraction of the cost of a minicomputer. An IBM PS/2 Model 80 with four Multiport/2s can control and monitor 31 different devices and report the status of those devices to a central system on the remaining port. As shown in Figure 1, computer systems that

Table 1 Comparison of processors

	PC XT	PC AT	M2	PS/2-30	PS/2-50
Microprocessor	8088	80286	80186	8086	80286
Processing speed (MHz)	4.77	6	7.37	8	10

Table 2 Memory comparison

	PC XT	PC AT	M2	PS/2-30	PS/2-50
ROM	64KB	64KB	16KB	64KB	128KB
Base RAM	128KB	256KB	512KB	640KB	1MB
Max RAM	640KB	15MB	512KB	640KB	7MB

now have mainframes controlling minicomputers which, in turn, control devices can be extended or scaled down using the Multiport/2.

Acting as a coprocessor or slave processor, the Multiport/2 can also be used to offload non-communications-related processing from its host PS/2. Its on-board 80186 microprocessor provides a processing capability comparable to that of a PS/2 Model 30.

Hardware

The major components of the Multiport/2 are the microprocessor, memory, serial I/O ports, custom gate array, and support microcode. The first three are comparable to major components of personal computers; the last two are unique. A discussion of each of these components follows.

Microprocessor. The 80186 microprocessor used by the Multiport/2 is a superset of the 8086 and 8088 microprocessors. The instruction sets are compatible, but only the 80186 is an integrated package that includes a programmable interrupt controller, three programmable timers, and two Direct Memory Access (DMA) channels. The instruction set of the 80186 microprocessor is a subset of the instruction sets of the 80286 and 80386 microprocessors used in the PS/2 Models 50, 60, and 80. While providing a programming environment compatible with its host PS/2, the 80186 satisfies the physical requirement to use as little surface area as possible. At 7.37 MHz, the processing speed of the Multiport/2 is comparable to that of the PS/2 Model 30. The processors

and processing speed of the Multiport/2 and several IBM microcomputers are compared in Table 1.

Memory. Like most other microcomputers, the Multiport/2 has both read-only memory (ROM) and dynamic memory, also called Random Access Memory (RAM). On system startup, the control microcode in ROM performs a power-on self-test and initializes all the components of the Multiport/2. Custom applications operating under the control microcode of the Multiport/2 reside in RAM and use the RAM for buffering data from the host PS/2 and serial I/O ports. Table 2 shows a comparison of the ROM and RAM on the Multiport/2 and other IBM microcomputers.

In addition to supporting applications and buffering data, the RAM on the Multiport/2 is the mechanism for communicating with the PS/2. Blocks of the Multiport/2 memory, called "pages," can be mapped into the PS/2 memory space so that programs running in the PS/2 can read and write the page of shared memory as if it were PS/2 memory. The location of this block of shared memory can be configured and placed in any unused block of memory below 16 megabytes within the PS/2's addressable memory space, thus creating a window into the Multiport/2's memory. In a typical application, this window would remain fixed within the PS/2 memory space; however, any page of the Multiport/2's RAM can be viewed. Each Multiport/2 in a PS/2 requires its own window within that PS/2's memory space. This may be thought of as being analogous to the windows in a house: The location of each window is chosen when the house is built; although windows can be moved, they typically are not. From inside the house, the view through a window changes when one looks in a different direction. Looking through different PS/2 windows allows the RAM of different Multiport/2s to be viewed. Altering the page to be shared in a window permits different pages of a Multiport/2's RAM to be viewed.

The placement of the window and the page of the Multiport/2 RAM being shared is not meaningful to applications running in the Multiport/2 because only the Multiport/2's memory space is available to them. Applications running in the PS/2 may send data through the window by choosing a page of the Multiport/2 RAM to be shared and then writing the data into the window. Receiving data is accomplished by choosing a page to be shared and reading the data from the window.

Serial I/O ports. On the Multiport/2, either four or eight serial I/O ports can be provided by an optional

electrical interface board (EIB). The first two ports are programmable in order to support asynchronous, byte-synchronous, and bit-synchronous protocols with external and internal clocking and a wide range of protocol-specific options. The remaining ports do not have the clocking inputs and outputs and, consequently, are asynchronous ports. All ports support programmable modem control inputs and outputs such as Data Terminal Ready (DTR), Data Set Ready (DSR), Ring Indicate (RI), Request to Send (RTS), and Clear to Send (CTS), depending on the electrical interface board installed. All ports may be configured to support variable frame sizes, stop bits, parity, and bit rates. The configuration of each port is independent of the configurations of all other ports, permitting one port to be programmed and operated for SDLC, a second port to support BISYNC, and the remaining ports to be configured for asynchronous communication with six different combinations of frame sizes, stop bits, and parity. To configure each port, specific I/O addresses are written with the desired configuration data. The highlights of the configuration options for a port are shown in Table 3.

All ports may operate in polled or character-interrupt modes of I/O. In addition, DMA I/O is supported for the first two ports (ports 0 and 1). The two DMA channels of the 80186 processor can each be assigned to satisfy receive or transmit requests for the first two ports. By different assignments, it is possible to support any one of the following:

- Port 0 or 1 with full-duplex DMA I/O
- Ports 0 and 1 both transmitting in DMA mode
- Ports 0 and 1 both receiving in DMA mode
- Port 0 transmitting in DMA mode with Port 1 receiving in DMA mode
- Port 1 transmitting in DMA mode with Port 0 receiving in DMA mode

Table 4 summarizes the capabilities of the ports.

All options for the serial ports are programmable. For each port, there is a block of sixteen I/O addresses that are used to specify the options selected for that port. Not all options are valid with all ports. For example, the DMA options are only valid for ports 0 and 1. By writing the desired values for a port to that port's block of I/O addresses, an application can configure the port as needed for that application. A summary of the block of I/O addresses for each port is shown in Table 5.

The electrical interface for all ports is chosen by an optional EIB which also has the external connector

Table 3 Port configuration options

Asynchronous Options (all ports)
5, 6, 7, or 8 bits per character
1, 1.5, or 2 stop bits
Odd, even, or no parity
Break generation and detection
Parity, overrun, or framing error detection
Byte-Synchronous (BISYNC) Options (ports 0 and 1 only)
Internal sync
1 or 2 sync characters
Auto sync insert and delete
CRC generation and checking
6- or 8-bit sync characters
Bit-Synchronous (SDLC/HDLC) Options (ports 0 and 1 only)
Abort sequence generation and checking
Auto zero insert and delete
Auto flag insert between messages
Address field recognition
CRC generation and checking
Bit Rates: Programmable for rates between 50 bps and 38.4K bps driven by internal baud rate generator or, on ports 0 and 1 only, external clocking.

Reproduced by permission, © 1986 Zilog, Inc. This material shall not be reproduced without the written consent of Zilog, Inc.

Table 4 Overview of Multiport/2's serial I/O ports

	Ports 0 and 1	Ports 2-7
Asynchronous	Yes	Yes
Byte-Synchronous	Yes	No
Bit-Synchronous	Yes	No
Polled I/O	Yes	Yes
Interrupt I/O	Yes	Yes
DMA I/O	Yes	No
Modem Controls	Yes	Yes
External Clocking	Yes	No

for the serial ports. Isolating the electrical interface definition from the base adapter allows compatibility with a wider variety of electrical requirements. Consequently, the Multiport/2 can be electrically compatible with other IBM systems, such as the Series/1 or 4300, as well as non-IBM systems and devices. These pluggable EIBs also permit easy alteration of the electrical interfaces for the ports. Custom EIBs may be built to satisfy unique interface requirements. Because the external connector is on the EIB, custom EIBs need not conform to a standard connector but may have the connector interface that best satisfies the requirements of each EIB.

Three EIBs are currently available for the Multiport/2. The first of these defines four serial ports to be RS-

Table 5 Overview of serial port's configuration block

Address	Function
Base I/O address + 0	Command register for condition resets
Base I/O address + 2	Interrupt and DMA control
Base I/O address + 4	Define interrupt vector*
Base I/O address + 6	Configure receive parameters
Base I/O address + 8	Configure stop bits, character synchronization, clock mode, and parity
Base I/O address + 10	Configure transmit parameters
Base I/O address + 12	Sync character 1 or SDLC address
Base I/O address + 14	Sync character 2 or SDLC flag
Base I/O address + 16	Transmit data buffer
Base I/O address + 18	Master control and enable*
Base I/O address + 20	Miscellaneous transmit/receive options
Base I/O address + 22	Configure clock options
Base I/O address + 24	Time constant high
Base I/O address + 26	Time constant low
Base I/O address + 28	Configure autoecho, local loop-back, Digital Phase-Locked Loop
Base I/O address + 30	Interrupt enable for modem controls and error conditions

*Not to be written by application.

Reproduced by permission, © 1986 Zilog, Inc. This material shall not be reproduced without the written consent of Zilog, Inc.

232C. The second and third include components to support four additional serial ports and to define the electrical interface for all eight ports. The second EIB defines all eight ports to be RS-232C; the third EIB defines ports 0 through 3 to be RS-232C and ports 4 through 7 to be RS-422A.

The communication speeds supported are dependent on the electrical interfaces, the modes of I/O (polled, character-interrupt, or DMA), the protocols (asynchronous, byte-synchronous, or bit-synchronous), and the applications. With the available EIBs, it is possible to transmit and receive simultaneously on either port 0 or 1 of the Multiport/2 using DMA mode at 38.4K bits per second (bps). Another port can be operating concurrently at rates up to 19.2K bps full-duplex. It is also possible to program all eight ports to operate simultaneously at 9.6K bps full-duplex. Higher baud rates are possible with custom EIBs that define the electrical interface of port 0 or 1 to be a high-speed interface. (RS-232C is a low-speed interface that cannot be supported at rates higher than 38.4K bps even with the DMA capability of port 0 or 1.)

Gate array. A very-large-scale integrated (VLSI) gate array developed and manufactured by IBM controls the Multiport/2's access to the PS/2 Micro Channel™. Access to the Micro Channel permits communication between applications on the PS/2

and applications on the Multiport/2. The gate array, known as the "Shared Storage Interface Chip" (SSTIC), includes the following functions:

- Mapping the shared storage page of Multiport/2 RAM into the PS/2 window
- Generating interrupts to the PS/2 from the Multiport/2 upon request
- Generating interrupts to the Multiport/2's 80186 from the PS/2 upon request
- Arbitrating RAM access between the 80186 and PS/2 microprocessors.

The method for transferring data between applications on the PS/2 and applications on the Multiport/2 is based on the shared storage window. Because the Multiport/2's 80186 and the PS/2's microprocessor operate independently, both microprocessors can try to access the same memory simultaneously. The arbitration of RAM access performed by the SSTIC prevents both microprocessors from writing conflicting data to the same location or one microprocessor writing to RAM that the other is reading. The SSTIC does not prevent one microprocessor from overwriting the data written by the other microprocessor; such conflicts must be handled by application-specific software.

Interrupts generated to the PS/2 or Multiport/2 synchronize and control software. An interrupt to the Multiport/2 can notify a Multiport/2 application that there is a command or data available for it; an interrupt to the PS/2 can indicate the completion of a command, the availability of status or data, or an error condition. Each Multiport/2 requires one of sixteen I/O blocks reserved by the PS/2 for the Multiport/2. Each 8-byte I/O block is used to pass control information to the SSTIC and status information to the PS/2. Because the Multiport/2 is programmable, communication between applications on the PS/2 and applications on the Multiport/2 can be defined to satisfy the unique needs of the application. The shared storage window and interrupt capabilities provided by the SSTIC are tools used to accomplish this communication.

The SSTIC receives configuration data via the programmable option select (POS). The Multiport/2 conforms to the PS/2 architecture requirement prohibiting adapters from having jumpers or switches. Instead, POS options selected at setup time from the adapter descriptor file (ADF) are written to the PS/2 CMOS RAM. For each Multiport/2, POS defines the size of its shared storage window, the location of its

shared storage window within the PS/2 memory space, its base I/O address, the interrupt level on which it can interrupt the PS/2, and clocking options for ports 0 and 1. A summary of the POS options is shown in Table 6.

Microcode. The Multiport/2 microcode provides a real-time multitasking environment for the Multiport/2 applications tasks. The major functions of the microcode include

- System initialization
- Command support
- Memory management
- Resource management
- First-level interrupt handling
- Task control
- Intertask communication and control
- Communication between Multiport/2 tasks and PS/2 applications
- Miscellaneous services

When the system is started, the Multiport/2 microcode performs a power-on self-test and initializes all components of the Multiport/2. Any errors detected in the self-test are recorded. Components that do not perform correctly during self-test are not made available for use by applications. After the self-test, all components of the Multiport/2 are initialized.

The microcode can be configured to support a variable number of tasks, priorities, and certain resources. When these configuration parameters are provided by the DOS or OS/2 driver, the microcode defines and initializes its own data area. Control blocks are used for each potential task and priority. Because the number of tasks and priorities supported is configurable, only the required number of control blocks are reserved. Consequently, a smaller data area is required for the Multiport/2 microcode. After the initialization is complete, the Multiport/2 microcode waits for a command from a PS/2 application.

All commands to the Multiport/2 from PS/2 applications are received initially by the microcode. These may be handled by the Multiport/2 microcode or directed to the responsible task for handling. A limited set of commands supported by the microcode includes commands to load a task to Multiport/2 RAM, start a task, stop a task, and unload a task from Multiport/2 RAM. If the command is not for the Multiport/2 microcode, it is directed to the task identified in the command information. Tasks can accept or ignore commands from PS/2 applications,

Table 6 Summary of Programmable Option Select (POS) options

Register	Function
POS 0 and 1	Adapter identification
POS 2	Base I/O address, interrupt level, sleep enable
POS 3 and 4	Shared storage window location in PS/2 memory space
POS 5	Shared storage window size, channel check indicators

depending on the architecture of the specific application. For example, a subsystem of tasks operating on the Multiport/2 can define one task to manage all communications with the PS/2 applications. In that subsystem, only that task would accept commands from the PS/2 applications. All other tasks in that subsystem would provide services to and accept commands from other tasks in that subsystem.

Dynamic allocation and deallocation of RAM are supported by the Multiport/2 microcode. The block size of RAM can be specified by the requesting task to be any multiple of 16 bytes up to the largest available RAM block. To alleviate the system overhead of dynamic memory compaction, memory blocks are compacted only on deallocation and only with adjacent available blocks. No compaction of memory is performed to satisfy requests for RAM.

Resource initialization, allocation, and deallocation are provided for Multiport/2 tasks. The fixed requestable resources are memory blocks, two DMA channels, eight serial I/O ports, five hardware timers, and 112 software interrupt vectors. The configurable requestable resources are 1 to 255 software timers and 1 to 255 user queues. Hardware timers and software timers allow tasks to be interrupted at user-definable intervals. Hardware timers are programmable for intervals up to 35 milliseconds in increments of 543 nanoseconds; software timers are programmable for intervals up to 327.6 seconds in increments of five milliseconds. The software interrupt vectors can be owned by multiple tasks and provide a method for intertask communication and control. User queues are first-in-first-out (FIFO) queues used to pass blocks of data. The blocks of data, which are called "queue elements," can be any size between 4 and 65,535 bytes. The lengths of the queues are limited only by available memory.

Limited first-level interrupt handling is provided by the microcode. The application code is responsible

for servicing the interrupt and clearing any necessary interrupt conditions. Leaving this responsibility to the application ensures that only the necessary overhead is performed and provides the real-time performance that applications may need.

Task control includes the dispatch of tasks according to priority. Tasks that have the same priority are given control in a round-robin dispatch. Any task

**The primary component of the
Realtime Interface Co-Processor
OS/2 Support package is the device
driver.**

can cause any other task to be stopped, started, dispatched, suspended, resumed, or unloaded. All of these services are provided by the microcode.

The combination of user queues and task control provides intertask communication and control. Intertask communication is also supported by the requestable software interrupt vectors. Each owner of a software interrupt vector may receive control to provide a requested service whenever any task causes that interrupt. This permits direct access between tasks without system overhead such as task dispatch.

A typical application consists of components residing in both the Multiport/2 and the PS/2. The Multiport/2-resident component consists of one or more tasks handling functions such as communications. The PS/2-resident component provides the user interface and secondary storage access. The microcode provides services and structures for communication between the Multiport/2-resident component and the PS/2-resident component.

System support software

System support software packages for both DOS and Operating System/2™ are supported for the Multiport/2. Because these two support packages are similar, only the OS/2™ package is discussed here. The

two major components of the IBM Realtime Interface Co-Processor OS/2 Support package are the device driver and the loader utility. Other components of the package provide aids for application development for OS/2 applications using the Multiport/2.

The primary component of the Realtime Interface Co-Processor OS/2 Support package is the device driver. (A device driver is the operating system extension that controls a particular type of device.) The Multiport/2 device driver provides support for PS/2-resident applications to interface with their counterpart applications in the Multiport/2. The real-mode environment of OS/2 is not compatible with communication applications. Consequently, the device driver for the Multiport/2 operates in the OS/2 protect-mode environment only. Consistent with the multitasking environment of OS/2, multiple applications under OS/2 may communicate through the device driver to multiple tasks on one or more Multiport/2s. The device driver controls access to the shared storage window by serializing requests from applications that require direct access to the window. Applications which do not require direct window access can request that the device driver write data to or read data from specific tasks using the structures provided by the Multiport/2 microcode. After installation, the Multiport/2 device driver remains resident in the PS/2 memory and occupies approximately 20K bytes of storage.

The loader utility of the Realtime Interface Co-Processor OS/2 Support package provides a tool needed by all Multiport/2 applications. It is used to move Multiport/2-resident applications from PS/2 secondary storage to Multiport/2 memory upon request. While moving the file to Multiport/2 memory, the loader utility handles any necessary relocation and can also be used to start the loaded application.

Development tools

Applications using the Multiport/2 have two components: A system component resides in the PS/2 and provides the user interface and system support; an adapter component resides in the Multiport/2 and can handle all other processing. Although a specific header format is required for the system component, all development can be performed in any PS/2 or IBM PC using standard compilers or assemblers. Applications that are to operate in the Multiport/2 may be programmed in Assembler or C. The IBM Realtime Interface Co-Processor C Language Support package provides libraries and rou-

tines necessary to write Multiport/2-resident applications in C.

Special tools for communication testing or real-time analysis may be helpful when debugging time-critical sections of the application. For early debugging and testing, the IBM Realtime Interface Co-Processor Developer's Kit provides a software-debugging utility for debugging Multiport/2-resident tasks. This debugging utility supports breakpoints, register display and modify, memory display and modify, program trace, and other functions typically required for early code development. The Developer's Kit also provides high-level language support for interfacing with the DOS or OS/2 driver, source-language sample programs, and a developer's guide.

Summary

Many applications need the capabilities of the IBM Realtime Interface Co-Processor Multiport/2. A PS/2 system with OS/2 has more capabilities than some applications will ever need. But for those applications that require real-time response, multiprotocol support, nonstandard communications support, or large numbers of communication ports, the Multiport/2 can supplement the environment provided by OS/2 and the PS/2 offering function in a microcomputer that could previously be had only in a minicomputer.

Personal System/2, PS/2, and AT are registered trademarks, and Micro Channel, Operating System/2, and OS/2 are trademarks, of International Business Machines Corporation.

Sandra C. G. Sykes *IBM Application Systems Division, P.O. Box 1328, Boca Raton, Florida 33432.* After joining IBM in 1981, Ms. Sykes worked as a diagnostic programmer on the Series/1. Now a staff programmer, she has been a member of the team which developed the Realtime Interface Co-Processor family of products since its inception in 1983 and is manager of the department responsible for firmware support of those products. Ms. Sykes holds Bachelor's degrees in mathematics and computer science from Florida Atlantic University in Boca Raton, where she is currently completing work on her Master's of Applied Science in computer and information systems.

Reprint Order No. G321-5319.