Volume 4, Number 1
IBM Personal Computer XENIX, Version 2.00

March 1986

IBM Personal Computer Seminar Proceedings

**The Publication for Independent Developers
of Products
for IBM Personal Computers**

**Published by International Business Machines Corporation
Entry Systems Division**

Changes are made periodically to the information herein; any such changes will be reported in subsequent Proceedings.

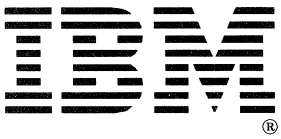
It is possible that this material may contain reference to, or information about IBM products (machines and programs), programming or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming or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

IBM believes the statements contained herein are accurate as of the date of publication of this document. However, IBM makes no warranty of any kind with respect to the accuracy or adequacy of the contents hereof.

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

Printed in the United States of America



Contents

Introduction and Welcome

These are the Proceedings of the IBM Personal Computer Seminar, designed for independent developers of products for IBM Personal Computers. The purpose of these Proceedings is to aid you in your development efforts by providing relevant information about new product announcements and enhancements to existing products. This issue is prepared in conjunction with this seminar. The Proceedings of future seminars for IBM Personal Computers also will be published and will cover topics presented at those seminars.

Throughout these Proceedings, the terms Personal Computer and family of IBM Personal Computers address the IBM Personal Computer, the IBM Personal Computer XT, the IBM PC*jr*, the IBM *Portable* Personal Computer and the IBM Personal Computer AT.

The IBM Personal Computer XENIX Operating System (referred to as PC XENIX), Version 2.0, which is discussed in this issue of the Proceedings, executes on the IBM Personal Computer (PC) AT.

Purpose

What is our purpose in issuing a publication such as this? It is quite simple.

The IBM Personal Computer family is a resounding success. We've had a lot of help in achieving this success, and much of it came from the independent developers.

As you proceed with your development, do you at times wish for some bit of information or direction which would make the job easier? Information which IBM can provide? This is the type of information we want to make available to you.

Since we want to be assured of giving you the information you need, we ask you to complete the questionnaire which appears at the end of these Proceedings. Your response to this questionnaire will be taken into account in preparing the content of future issues, as well as the content of seminars we will present at microcomputer industry trade shows.

Topics

The following list gives a general indication of the topics we plan to cover in future seminars and include in the IBM Personal Computer Seminar Proceedings:

- Information exchange forum — letters to the editor format
- Development tools — languages, database offerings
- Compatibility issues
- New devices — capacities and speeds
- System capacities — disk and memory
- Enhancements in maintenance releases
- Tips and techniques
- New system software
- Hardware design parameters
- Tips on organizing and writing documents for clear and easy reading
- Changes to terms and conditions

IBM Personal Computer XENIX, Version 2.00

Overview

IBM Personal Computer (PC) XENIX®[1] Version 2.00 is an operating system for the IBM Personal Computer (PC) AT which utilizes the advanced features of the 80286 processor. The PC XENIX system brings many of the features normally found on mainframe computers to the IBM Personal Computer AT.

The PC XENIX operating system is a multitasking, multiuser system derived from AT&T UNIX[2] System V Operating System, Release 1. "Multiuser" on the IBM PC AT means that up to three users can use the system at the same time. Two users can access PC XENIX through ASCII terminals or PCs and a third user can access PC XENIX using the system console. The multiuser capability also allows multiple usage of the same file or program. This capability makes possible the immediate updating and sharing of critical data from several sources. PC XENIX file management controls individual file execution permissions, read permissions and write permissions for file security and integrity. Multitasking on the IBM PC AT means the concurrent running of several programs, including the programs of one or more users. The PC XENIX system allows programs and tasks to be initiated as background processes, freeing the terminal to do other work while a time consuming task is being completed in the background. This increases productivity because tasks are run simultaneously rather than in sequence.

The PC XENIX system is comprised of four separate program products. The Operating System is the base package. It requires at least an IBM Personal Computer AT with 512KB of memory, 20MB fixed disk drive, a 1.2MB diskette drive and a display. The Operating System is a prerequisite for the Software Development System, the Text Formatting System and the Operating System Extensions.

The Software Development System contains a large set of application software development tools, including the C programming language. Together, the C compiler, 80286 Macro Assembler, interactive debugger, productivity tools and other programming utilities provide a complete environment for the programmer.

The Text Formatting System provides high-level formatting macros for document preparation in conjunction with nroff (a formatter for printers) and troff (a formatter for phototypesetters). Additional formatting facilities are also provided.

The Operating System Extensions include many enhancements from INTERACTIVE Systems Corporation. It provides additional end-user functions in the PC XENIX environment which enhance compatibility with other IBM UNIX derivative products. It includes a full-screen editor with windowing, a function-key driven user interface, UNIX System V accounting functions and versatile inter-system communication facilities.

Operating System

The Kernel

The Operating System is composed of many parts. One of these parts is the PC XENIX kernel. The kernel is the central part of the operating system. It controls all the resources of the computer and allocates them to the active processes. The kernel manages the file system, automatically assigning disk space to files and directories as they are created or enlarged, and returning no longer used space to the "free list".

The kernel takes advantage of the protect mode of the IBM Personal Computer AT 80286 processor to protect itself from the users and to protect the users from each other. It controls access to the system and to the files through password protection and file permissions. Each user is granted access only to those files and directories that he or she is authorized to use.

Commands and Usage

Over 150 commands are available with the PC XENIX Operating System. They include two editors: "vi" and "ed". Vi is a full-screen editor which provides many commands to insert, replace, move and search for text. Ed is a line editor that is available in most UNIX-derived systems. These commands create and maintain files and directories, create file systems, mount and unmount file systems, link files across directories and

1 XENIX is a registered trademark of Microsoft Corporation
2 UNIX is a trademark of AT&T Bell Laboratories

communicate with other users. The capabilities of linking and combining these commands with shell scripts, redirection and pipelines are among PC XENIX's most powerful features.

PC XENIX offers you a choice of three shell command interfaces to the system. You can select the shell to best suit your needs and experience:

Bourne Shell

The Bourne shell (sh) is a command language interpreter that serves as the interface between the user and the rest of the system. It interprets commands, calls the corresponding programs into memory and executes them. The shell serves as both an interpreter and a programming language. This permits the creation of new commands simply by writing shell scripts (also called procedures). The shell language contains flow control constructs such as while, if, else, for, do and case. It also allows the definition of variables, parameter passing and the processing of software interrupts.

A shell script is a file that contains a collection of PC XENIX commands, pipelines, shell conditional and flow control constructs, and parameter names. The procedure is invoked by simply typing its name optionally followed by one or more arguments.

Visual Shell

The PC XENIX Visual Shell (vsh) is a full screen shell interface that optionally replaces the Bourne shell as the command interpreter. It is recommended for casual users of the system and those with specialized needs. The Visual Shell replaces the operating system command line with a visual menu of the most commonly executed applications and utilities and provides help information for each selection. The user can select a file for use with a command simply by pointing to it via cursor movement keys.

Another important feature of the Visual Shell, especially valuable to software developers, is the ability to easily customize the shell. By developing appropriate shell scripts or programs and changing the words on the menu, application developers can make a simple application interface for the occasional system user. This flexibility enables complicated data processing tasks to be performed with minimal computer knowledge.

C Shell

The PC XENIX C shell (csh) is a command language interpreter. Like the PC XENIX Bourne shell (sh), the C shell is an interface between the user and the operating system, translating command lines typed at the terminal into corresponding system actions. It is packaged as part of the IBM PC XENIX Operating System.

The C shell offers features above those provided by the Bourne shell. The C shell can maintain a history list, into which it places the text of previously entered commands. Using a special notation, one can reuse commands or words from previously entered commands to form new commands. This mechanism is useful for repeating previous commands or correcting typing mistakes.

The alias facility simplifies commands, supplies default arguments, and performs transformations on commands and their arguments. This mechanism enables the user to substitute a personally preferred name for a system command, for a system command and its arguments, or even to represent multiple commands or pipelines.

Also provided by the C shell are language structures similar to those of the C language, including while, foreach, if, else and switch.

Input/Output Redirection

The PC XENIX shell structure provides a flexible means for redirection of standard input and output. The standard input is the keyboard and the standard output is the display. By using this feature, each physical I/O device, including the keyboard, the display and other peripherals can be treated as any normal file. Thus, via a shell, the input to a command can come from the keyboard, from a file stored on a disk or from another program and still look like the standard input. Similarly, the output from a command can be redirected from the display to the printer, to a disk file or to another command. With these capabilities a single PC XENIX command can often perform several distinct tasks. For example:

```
cat x y > z
```

concatenates files x and y and puts the result in file z.

Pipelines

PC XENIX has the ability to pipe data from one command to another. Two or more connected programs are called a pipeline. The shell can execute a sequence of commands, with the output of each command becoming the input to the next command in sequence. This ability to pipe commands often eliminates the need to develop new programs for new applications. For example, if an application requires sorting a file of words, removing all duplicate entries, counting the number

of remaining words, and printing the results, four existing PC XENIX programs can be piped together to perform this new action:

```
sort words | uniq | wc | lpr
```

The vertical bar is the pipe symbol. The meaning of this pipeline is (1) sort the file named words, (2) run the sorted list through the uniq program, which removes duplicate entries, (3) run the output through the wc program, which counts the number of words, and then (4) print the output of wc on the printer using the lpr command.

The PC XENIX File System

The PC XENIX file system consists of files and directories arranged in a tree-like, hierarchical structure. It allows filenames up to fourteen alphanumeric characters in length and differentiates between uppercase and lowercase. The PC XENIX file system, which is a hierarchical file system, consists of many levels or hierarchies, i.e., directories within directories.

The file system has a unique path for each file in the system. This unique path is the name of the file in relation to the root of the file system. The root of the file system is the parent of all other members of the file system and is designated by the slash symbol (/). Within the file system some directories are standard and contain PC XENIX command files. The directory /bin contains most of the common commands. The directory /etc has the super-user (system administrator) commands. There are other standard directories such as /usr, where user directories and files are stored.

Each user of the PC XENIX system is assigned a login or home directory. When a system user logs in using the assigned login name, he is automatically in his own directory, which is a directory with the same name as his login name in the /user directory. Starting at his home directory, a user can create and delete files and directories as needed.

Editing Facilities

Two editors are included in the PC XENIX Operating System. A third editor, provided in the Operating System Extensions, will be discussed later. Together, these editors provide varied approaches to editing because they are designed to suit the needs of two groups: the programmer who prefers line editing and the user who prefers a full screen editor. Any ASCII text file can be created and modified using one of two text editors, ed and vi, available with the PC XENIX Operating system.

Communications

Among the several ways to communicate with other users on the PC XENIX system are the two basic facilities, write and mail.

The write command allows communication with any user currently logged on the system. One user can write to another user, then wait for the other user to write back. Following the execution of the write command, whatever is typed on the first user's terminal appears on the other user's terminal and vice versa. The users take turns reading and writing messages until one decides to end the conversation.

The other basic way to communicate with another user is with the mail command. Mail sends a message or a previously created file to the other user. The receiving user does not have to be logged on or even be a user on the same system to receive mail. Mail is delivered to the user's mailbox directory and when the receiving user logs in, a message indicates that mail has arrived. The receiving user can then read, delete, save or transfer the received mail to other users.

Networks

If you need to communicate with a receiving user who is not on the same system (i.e., the login or home directory is in another system), additional communication facilities may be needed. First, the two systems must be able to communicate with one another. If the systems are not far apart, e.g., across the room, RS232 lines and the serial ports on the IBM Personal Computer AT can be used to form a network. If the other system is across town or across the country, a modem and the uucp communication system can be used to complete the network.

In PC XENIX, the uucp command retains its UNIX meaning of UNIX-to-UNIX copy. IBM PC XENIX also has micnet which is a limited local network that connects personal computers via the serial ports. Once a form of communication has been established between the two systems, the mail command is used to communicate with users in the other system. Commands within micnet allow copying or transferring files and accomplishing tasks such as executing commands on a remote PC XENIX system. The uucp communication system can combine the usefulness of local micnet systems by connecting them via modems.

Co-Residence of Multiple Operating Systems

Some PC XENIX users may wish to use another PC operating system. PC XENIX is designed so that part of the fixed disk may be reserved for another operating system, such as IBM Personal Computer DOS. During installation, PC XENIX allows one to specify a disk partition for PC XENIX that is less than the full size of the disk, so that subsequent installation of another operating system on the remainder of the disk is possible. PC XENIX requires a minimum of three partitions.

If you intend to install multiple PC XENIX software products on your PC AT, you should limit the amount of disk space you allocate for an operating system partition accordingly. For example, with the Operating System, Software Development System, Text Formatting System and Operating Systems Extensions all installed on a 20M byte fixed disk, approximately 17.5M bytes are utilized. This leaves approximately 2.5M bytes for an additional operating system.

PC XENIX provides the capability to transfer files between PC DOS diskettes or fixed disk partition and the PC XENIX file system. This capability is discussed later.

Administrative Utilities

PC XENIX has several super-user commands that facilitate the maintenance of the users' logins. The system administrator is the super-user and is the only one permitted to use these commands. The "mkuser" command is used to add a user login to the system. This command handles all directory creation, shell assignment and password updating. When the super-user creates a user login, he assigns one of the shells (Bourne shell, C shell or Visual shell) as the default shell. The "rmuser" command does the opposite. It makes sure the user's mailbox is empty and that all files belonging to that user have been deleted. Rmuser then deletes the user's name from the password file and removes his home directory. The "chsh" command allows the super-user to change a user's default shell.

PC DOS Utilities

PC XENIX provides the capability to transfer files between IBM Personal Computer DOS diskettes or DOS fixed disk partition and the PC XENIX file system. There are several utilities to access DOS files and directories on DOS diskettes or fixed disk. The commands have a "dos" prefix. They include support to copy files, list files and directories, erase files, and create and delete directories.

One of the PC XENIX commands to access DOS files is doscp. It copies files between a DOS diskette and the PC XENIX file system. Doscp takes two arguments, file1 and file2, and copies the contents of file1 to file2. Either filename can be a DOS file by using the following form:

```
device:file
```

where "device" is the path name of the device containing the DOS file.

A simplified way to specify the DOS device is provided in PC XENIX via a set of capital letters that represent the DOS devices. For example, to copy a DOS file named file1 from a DOS diskette in drive A to the PC XENIX file system and keep the same filename, the syntax would be:

```
doscp A:file1 file1
```

To copy file1 from the PC XENIX file system to a DOS diskette in drive B and rename file1 to file2, the syntax would be:

```
doscp file1 B:file2
```

Terminal Support

Termcap is a PC XENIX data base describing terminals. It is used by programs such as vi to interface with a user's terminal. IBM PC XENIX termcap supports many ASCII terminals, including those manufactured by vendors other than IBM. Termcap is packaged as part of the IBM PC XENIX Operating System.

Some of the terminal features that are described in a termcap data base entry include the number of lines, the number of columns on each line, how the screen is cleared, backspace capability, cursor motions, highlighting, underlining and insert/delete character keys. Termcap support can also be easily added for new terminals as they become available.

As a complement to termcap, curses is a screen updating and cursor movement set of library functions that are packaged as part of the IBM Personal Computer XENIX Software Development System. Curses relies on the information in the termcap data base to tailor terminal commands allowing the user to write terminal-independent programs. These functions provide a simple and efficient way to use the capabilities of the terminal attached to the program's standard input and output files.

Software Development System

The PC XENIX Software Development System contains a large set of software application development tools including a C compiler, a 80286 Macro Assembler, interactive debugging and such productivity tools as the command "make" and the Source Code Control System (SCCS).

The C Language and Assembler

C is a general-purpose programming language. Most of the software in the PC XENIX system is written in the C programming language.

The C compiler can create programs for three different types of memory models: Small model (Ms), Middle model (Mm), and Large model (Ml). Additional model support (Mh) is provided to handle arrays that exceed 64K bytes.

Each type of model is distinguished by its use of available memory space and how it uses programs and data in that space.

Small model programs can be pure or impure in regard to space usage. Impure-text, small-model programs occupy one 64K byte physical segment. The program and data areas are combined in one segment. Pure-text, small-model programs occupy two 64K byte physical segments. One segment is used for data. The other segment is used for text (program) and is read-only. This means that a pure-text program can be shared by several processes at a time (only one copy is loaded in memory). The total program size for a pure-text small-model is 128K bytes.

Middle-model programs use one segment for data and as many segments as necessary for the program. There is no limit on the program (text) size. The only limitation is that the space required for variables (data) does not exceed 64K bytes.

Large-model programs use as many segments as necessary for data and programs. The program, however, must not contain a single array or structure exceeding 64K bytes.

A significant feature of the IBM Personal Computer XENIX C compiler is DOS cross compilation. This feature lets the programmer create code to be run under IBM Personal Computer DOS (Version 2.00, 2.10, 3.00 or 3.10). The programmer creates and debugs C programs using the productivity tools available in PC XENIX, then recompiles the programs for PC DOS. PC XENIX provides a library for PC XENIX programs and another library for PC DOS C programs. The C compiler selects the correct library and generates the appropriate load module. The programmer can then use PC DOS utilities available in PC XENIX to copy the load module to a PC DOS diskette or DOS fixed disk partition.

Sometimes certain tasks cannot be done in C or will be better performed when written in assembler language. The PC XENIX Software Development system provides an 80286 macro assembler.

Application Development Tools

Medium to large programming projects tend to have multiple source files. There may be too many dependencies between these files to be remembered at all times. A modification in one file might affect other files. The "make" command is a program maintainer that coordinates these dependencies. It reads a file supplied by the programmer which specifies the file dependencies in the project. Usually a programmer specifies that a load module depends on certain object modules, and each object module depends on a source file. If a source file has changed (its date is newer than the date for the object file) it causes the source file to be recompiled. The new object file is then linked with other object files to produce a new load module. This way the programmer does not have to remember the files that have to be recompiled and relinked when a change occurs.

The Source Code Control System (SCCS) controls the maintenance of any text or source code files. The maintenance process is started by running the SCCS "admin" command on the files containing the text. SCCS provides other commands to retrieve and update the files, add new releases, and recreate the complete source of a release at any point in time. SCCS saves the original files and maintains a list of modifications (called deltas) that have been applied to each of the files. To obtain a specific release, SCCS starts with the original file and applies the correct deltas to the files involved.

The PC XENIX Software Development System contains additional commands including an archiver, a C beautifier, a lexical analyzer called "lex", preprocessors and a compiler-compiler.

Text Formatting System

The IBM Personal Computer XENIX Text Formatting System is composed of a collection of tools that complement writing productivity. In combination with any of the system's text editors, the text formatting system can be especially valuable in simplifying the generation of technical reports, formal papers and other professional quality documents.

Most of the commands to format text appear at the beginning of text lines and consist of one or two letters preceded by a dot. The desired output is generated by executing a formatter program that uses a text line as input to create the desired results.

The main components of the Text Formatting System are the (nroff) and (troff) formatters. The nroff formatter produces output for matrix line printer devices. The troff formatter, compatible with nroff, offers more sophisticated capabilities because it provides output suitable for phototypesetters. The output of nroff and troff can be transmitted from the Personal Computer to computers equipped with printers and typesetters that can print such documents. For example, nroff uses underlining, approximate spacing and text size determined by the characteristics of the printer used; troff supports italics, variable spacing and point size.

Additional formatting facilities may be used in conjunction with nroff and troff for building mathematical equations and for designing complex tables. The equation (equ) preprocessor can format complicated mathematical symbols and equations using commands that resemble English words. The (tbl) preprocessor gives control over material that must appear in tabular form, and it automatically calculates the information necessary to line up complicated columnar data even if elements have varying widths. It also centers tables, expands a table to the width of the current line and encloses a table in a box.

Because the text formatting package is a set of building blocks that work together, the user can combine formatting commands like equ, tbl and troff into one file and process it all at once.

Text Analysis

Another series of text formatting system utilities for analyzing writing style and correctness are style, diction and explain. The style utility analyzes writing style and reports on readability, sentence length and structure, word length and usage, verb type and sentence openers. The diction utility finds all sentences in a document that contain phrases from a data base containing bad or wordy diction. The explain utility interactively suggests alternatives to phrases flagged by the diction utility.

The PC XENIX spell utility is useful in finding misspelled words in a file. It contains a dictionary of words and it supports adding additional words. It also has an option that checks for British spellings such as colour and centre.

Operating System Extensions

The IBM Personal Computer XENIX Operating System Extensions provides additional end user functions in the IBM PC XENIX environment. These added functions are designed to enhance compatibility between IBM PC XENIX and other IBM UNIX derivative products. The PC XENIX Operating System Extensions features the TEN/PLUS[3] function-key driven user interface, a full screen communication facilities, PC/IX file migration aids and UNIX System V accounting functions.

User Interface (TEN/PLUS)

TEN/PLUS provides a full screen, function-key driven user interface for the IBM PC XENIX Operating System. Built around the INed[3] editor, TEN/PLUS features mail management, file management and history management. Mail management facilitates the selecting, reading, creating, filing and editing of messages. File management facilitates selecting, reading, creating, moving, deleting, copying, linking, editing and other file related operations. History management facilitates maintaining and recreating the history of changes made to INed files. The TEN/PLUS user interface also provides user definable "pop-up" menus and helps.

INed Editor

The INed editor is a text oriented, full screen editor featuring multiple windows with both window and cursor positioning. Corrections, insertions and deletions are made by cursor positioning and typing. The INed editor provides move operations that allow for the movement of text within a file or from one file to another. Vertical and horizontal scrolling are provided to allow moving through a file or viewing lines that are longer than the screen width. The editor does automatic scrolling on input permitting additional input while retaining some lines for context.

Communications Facilities

IBM PC XENIX is capable of asynchronously communicating with other UNIX systems over communication lines (direct connect, leased lines or dialed lines). The Operating System Extensions enhance this support through the Connect facility[3], FTP Program, INmail[3] and INnet[3]. Connect is a facility to connect to a remote UNIX system, or any other system supporting the same ASCII protocol as a UNIX system, and appear to that system as a

[3] TEN/PLUS and INed are registered trademarks of INTERACTIVE Systems Corporation
INmail, INnet and Connect are trademarks of INTERACTIVE Systems Corporation

terminal. FTP is a program that permits a user to access the file system of a remote host, and executes file-oriented commands interactively. INmail is a message sending and receiving facility that incorporates prompts and uses private mail boxes. All facilities for sending and receiving messages are supported using the INed editor. INnet is an extension of INmail that provides communication between other processors running PC XENIX, IX/370, AIX or PC/IX with INmail/INnet/FTP. INnet also has an interface to uucp.

PC/IX File Migration Aids

The PC/IX file migration aids assist the user in moving files from PC/IX to PC XENIX. Programs are provided for restoring the contents of a PC/IX dump volume into a PC XENIX filesystem (pcixrestore), reading and copying a PC/IX filesystem on diskettes (pcixread) and listing a PC/IX filesystem on diskettes (pcixls). PC/IX restore reads data which is in dump format as created by the PC/IX dump command. Pcixread and pcixls read data which is in PC/IX filesystem format.

UNIX System V Accounting Functions

UNIX System V accounting functions are provided through PC XENIX Operating System Extensions. The accounting functions include an adaptable reporting system which details system usage and reports a chronological history of terminated processes that includes user and system resources, start and end times, owner associated with process, and system exit status.

The reporting functions provide connect-time accounting, as well as command usage, command frequence, disk utilization and line usage data. All of the accounting measurements can be summarized by user name and by command on a daily, monthly and fiscal basis.

Operating Environment

IBM Personal Computer XENIX Operating System has the following hardware configuration requirements:

- IBM Personal Computer AT
- 512KB RAM Memory
- One IBM high-capacity 1.2MB diskette drive
- IBM 20MB fixed disk
- One IBM display and adapter

IBM Personal Computer XENIX Operating System is licensed for a maximum of eight users; however, the existing IBM PC AT hardware will support only three concurrent users. The serial port on a serial/parallel adapter card is required to attach an asynchronous ASCII terminal for each additional user terminal. A maximum of two serial/parallel cards may be installed in an IBM Personal Computer AT system unit, thus a maximum of three users may concurrently run PC XENIX. The customer is responsible for the selection and installation of all hardware and software past the end of the IBM communication cable attached to the serial/parallel card. The following devices are supported:

- 80287 Math Co-processor
- 360KB diskette drive
- 30MB fixed disk
- IBM 3161 and 3163 (in 3161 mode) ASCII Display Terminals
- IBM 5151 Monochrome Monitor and adapter
- IBM 5153 Color Monitor and adapter
- IBM 5154 Enhanced Color Display and adapter
- IBM 5175 Professional Color Display and adapter
- IBM Proprinter
- IBM 5182 Color Printer
- IBM Graphics Printer

IBM PC XENIX Open Architecture

When IBM Entry Systems Division announced the IBM Personal Computer, it established a philosophy of open architecture. The intent was to encourage both hardware and software developers to develop products for the family of IBM Personal Computers.

The open architecture philosophy was a design goal of IBM PC XENIX. PC XENIX provides a number of well-defined interfaces which allow independent developers to enhance the basic product offering. It is also possible to tailor the PC XENIX kernel to the specific needs of a particular market sector. The following sections describe the various aspects of the PC XENIX open architecture in more detail.

Reconfigurable Kernel

The IBM PC XENIX kernel can be reconfigured in various ways to more specifically address the needs of the user. The following list of configurable items serves as an indication of the level of configurability designed into the PC XENIX product:

- PC XENIX Kernel Size
- Size of Kernel Buffers
- Number of Concurrent Processes
- Number of Open Files Per Process

PC XENIX is designed to allow users to quickly and easily modify the parameters which define each of these kernel features.

Device Drivers

The most flexible feature of PC XENIX is its ability to conform to a wide range of user-specific hardware. This is accomplished through a kernel module called a device driver. A PC XENIX device driver is a software interface that communicates with a hardware device and provides a means by which PC XENIX can control the device to perform input and output operations.

PC XENIX is shipped with the following device drivers for PC AT:

- Keyboard Device Driver
- Display Device Driver
- Asynchronous Communications Device Driver
- Floppy Disk Device Driver
- Fixed Disk Device Driver

Device drivers may be written to provide an enhanced level of function over that shipped with the product. User supplied device drivers must be incorporated into the PC XENIX kernel.

PC XENIX supports linkable device drivers and installable device drivers. The PC XENIX system administrator can link linkable device drivers into the kernel by remaking the kernel. Installable device drivers can be loaded by the operating system during system boot. The installable device driver feature allows much simpler device installation procedures.

Once loaded, the installable device driver is installed into the kernel image in order to provide access to the device which it is designed to control. The creation of an installable device driver requires only a small additional effort over that required for ordinary drivers which are linked into the kernel image during a configuration process.

In order to create an installable device driver, the following data structures must be added to an existing device driver, or these data structures must be included when developing a new device driver:

struct iddsw	This data structure contains control information for the program which performs installation during bootstrap.
struct bdevsw	This data structure is included if the device driver is a block device driver (e.g., tape drive or disk drive).
struct iddsw	This data structure is included if the device driver is a character device driver (e.g., terminal or line printer).

After the installable device driver is written for the system, one can then prepare the device driver object module to be an installable image that the boot program can dynamically link to the kernel at bootstrap time. This produces an installable device driver.

Linkable device drivers are those that can be linked directly into the PC XENIX kernel. The PC XENIX Software Development System must be installed on the system to produce a linkable device driver. To build a linkable device driver, do the following steps:

1. Move the device driver source code to the /usr/sys/io directory.

2. Add the name of the device driver file objects to the OBJS list in the makefile in the /usr/sys/io directory.

3. Type "make" in the /usr/sys/io directory to create the device driver object files.

4. Modify the master file in the /usr/sys/conf directory to include information about the new driver.

5. Modify the xenixconf file in the /usr/sys/conf directory to include information about the new driver.

6. Build a version of the kernel containing the new drivers by typing "make" in the /usr/sys/conf directory.

7. Copy the new version of the kernel to the root (/) and enter it as the executable image to the boot program.

Device drivers are the link between the PC XENIX kernel and the external world. The capability to alter or enhance this connection provides the user with a high degree of flexibility. The guidelines for the development and installation of device drivers can be found in the *IBM PC XENIX System Administration Guide* and the *IBM PC Application Development Guide*.

PC XENIX and System BIOS

PC XENIX was developed to take full advantage of the IBM Personal Computer AT architecture. In order to accomplish this, it was necessary for PC XENIX to operate in protect mode of the Intel[4] 80286 which is the processor of the IBM Personal Computer AT. Utilizing this mode of operation prohibits certain real mode (Intel 8086)[4] code from executing properly.

To avoid confusion and complication in this area, it was decided to develop the system level BIOS in compatibility mode which would allow all DOS applications to operate properly. However, when executing PC XENIX the system level BIOS is only partially used. The following sections provide a better understanding of the issues concerning PC XENIX and BIOS.

Power-On Self Test (POST)

When the IBM Personal Computer AT is powered on, the code responsible for Power-On Self Test (POST) is performed. This code is resident in the BIOS ROM and ensures the proper functioning of the system. If particular hardware components are not attached or are not functioning properly, this code will alert the user and attempt to provide an error message which isolates the problem.

Since this code is executed prior to any other code, it is not necessary to be concerned with the mode in which this code executes. Likewise, it is not necessary to duplicate the functions of this code in the PC XENIX kernel. Therefore, some code savings are provided by allowing the system level BIOS to perform these power-on system tests prior to loading and executing the PC XENIX kernel.

Bootstrap Loader

The only other feature of the BIOS ROM that is used when PC XENIX is installed is the bootstrap loader, which is executed immediately after a successful POST. This code loads a few sectors of code from the active partition on the boot device and then allows that code to execute. The loaded code is typically an operating system-specific loader which will ultimately load the operating system.

The features provided by the ROM BIOS bootstrap loader allow the operating systems to be uniformly and properly loaded based on the user's definition of the active partition and the boot device.

[4] Intel is a trademark of Intel Corporation

Protect Mode

The Intel 80286 can operate in one of two distinct modes, real or protect mode. IBM Personal Computer XENIX executes in protect mode. The major obstacle which prevents real mode (standard DOS) programs from executing in protect mode of the Intel 80286 is the level of protection which is imposed by the processor during execution. The Intel 80286 provides the following four levels of protection while executing in protect mode:

- Restricted Use of Certain Instructions
- Separation of System and User Code
- Separation of Individual Users' Code
- Data Type Checking

The real mode of operation allows the 80286 to emulate the 8088/86 processors with some minor instruction differences. The protected mode of operation allows the 80286 to provide a variety of enhanced features. The following is a list of the type of features that the Intel 80286 processor provides while executing in the protect mode:

- Memory Management
- Memory Protection
- Increased Memory Addressing
- Enhanced Memory Addressing Modes
- Enhanced Processor Instruction Set

The two major features in this list, memory management and memory protection, are discussed in detail below to provide a better understanding of the advantages of program execution in protect mode.

Memory Management

The Intel 80286 while operating in real mode has the same limitations on memory addressing as the 8088/86. Simply stated, the address space is limited to 1M byte. However, due to the design of the IBM Personal Computer, the actual maximum application memory address size is limited to 640K bytes in order to allow space for memory mapped I/O devices.

Even though the IBM Personal Computer AT may have up to 3M bytes of memory, if the processor is operating in real mode, only the first 640K bytes are actually available for program execution. If the 80286 is operating in protect mode, as in the case of PC XENIX, the program addressability is much more than 640K bytes.

While operating in protect mode the processor can address up to 16M bytes of real memory and considerably more logical memory. Utilizing this mode, PC XENIX is capable of managing as much memory as the user is capable of installing in the IBM Personal Computer AT.

Memory Protection

There is no explicit memory protection while executing in the real mode of the Intel 80286. Therefore, it is not possible to restrict the following types of activities:

- Restricted Use of Instructions
- Protection of System Code
- Protection of Users' Code
- Data Type Enforcement

However, while executing in the protected mode of the Intel 80286, it is possible to provide the following four levels of protection at the system and the user level:

- Restricted Use of Certain Instructions
- Separation of System and User Code
- Separation of Users' Code
- Data Type Checking

UNIX System V/ PC XENIX Comparison

UNIX System V / PC XENIX Comparison

The following chart is intended to be used as a guide in comparing UNIX System V and PC XENIX functions. Although this chart is not a comprehensive list of all AT&T UNIX System V functions, it does completely list IBM PC XENIX functions. It is hoped that this chart will be helpful in understanding the differences and similarities between the two products. Note that in this chart, "V" is AT&T UNIX System V and "XENIX" is PC XENIX for the IBM Personal Computer AT. Note also, the PC XENIX program product designations in the XENIX column:

- "o" for the Operating System
- "s" for the Software Development System
- "t" for the Text Formatting System
- "e" for the Operating System Extensions

Commands/ Applications	V	XENIX
300	x	
4014	x	
450	x	
abt	x	
accept	x	o
acct	x	e
acctcms	x	e
acctcom	x	o
acctcon1	x	e
acctcon2	x	e
acctdisk	x	e
acctdusg	x	e
acctmerg	x	e
accton	x	o
acctprc1	x	e
acctprc2	x	e
acctsh	x	e
acctwtmp	x	e
acuset	x	
adb	x	o
admin	x	s
aliashash		o
ar	x	s
ar.pdp	x	
arcv	x	
as	x	s
as.pdp	x	
asa	x	
asktime		o
asm		s
assign		o
at		o
atb	x	
awk	x	o
backup		o
badtrack		o
banner	x	o
basename	x	o
batch	x	o
bc	x	o
bcopy	x	
bdiff	x	o
bfs	x	o
brc	x	
bs	x	
cal	x	o
calendar	x	o
cancel	x	o
cat	x	o
cb	x	s
cc	x	s
cd	x	o
cdc	x	s
cflow	x	
chargefee	x	e
checkall	x	
checkcw	x	t
checkeq	x	t
checklist	x	
checkmm	x	t
chgrp	x	o
chmap	x	
chmod	x	o
chown	x	o
chroot	x	o
chsh		o
ckpacct	x	e
clear		o
clri	x	o
cmos		o
cmp	x	o
col	x	t
comb	x	s
comm	x	o
config	x	o
connect		e
convert	x	
copy		o
cp	x	o
cpio	x	o
cpp	x	s
cprs	x	
crash	x	
cref		s
cron	x	o
crontab	x	o
crypt	x	
csh		o
csplit	x	o
ct	x	
ctags		s
cu	x	o
cut	x	t
cw	x	t
cwcheck		t
cxref	x	
daemon.mn		o
date	x	o
dc	x	o
dcopy	x	
dd	x	o

Commands/ Applications	V	XENIX
deassign		o
delrem		e
delta	x	s
deroff	x	t
devnm	x	o
df	x	o
dgn	x	
diction		t
diff	x	o
diff3	x	o
diffmk	x	t
dircmp	x	o
dirname	x	o
dis	x	
disable	x	o
dmesg		o
dodisk	x	e
don	x	
doscat		o
doscp		o
dosdir		o
dosld		o
dosls		o
dosmkdir		o
dosrm		o
dosrmdir		o
dpd	x	
dpr	x	
dskfmt	x	
dstart	x	
dtype		o
du	x	o
dump	x	o
dumpdir		o
e		e
echo	x	o
ed	x	o
edit	x	o
efl	x	
egrep	x	o
emulcntrl	x	
emulload	x	
emulstat	x	
enable	x	o
env	x	o
eqn	x	t
eqncheck		t
errdead	x	
errdemon	x	
errpt	x	
errstop	x	
ex	x	o
explain		t
expr	x	o
f77	x	
ff	x	
factor	x	o
false	x	o
fdisk		o
ffill		e
fformat		e
fget	x	
fgrep	x	o
file	x	o
filesave	x	
fill		e
finc	x	
find	x	o

Commands/ Applications	V	XENIX
finger		o
fjust		e
format	x	o
frec	x	
fsck	x	o
fscv	x	
fsdb	x	
fsend	x	
fsplit	x	
ftp		e
ftpmail		e
ftpsrvr		e
ftpuser		e
fts	x	
fuser	x	
fwtmp	x	e
gcat	x	
gcosmail	x	
gdev	x	
ged	x	
get	x	s
getopt	x	o
gets	x	o
getty	x	o
ghost		e
graph	x	
graphics	x	
greek	x	
grep	x	o
grpcheck		o
grpck	x	o
gutil	x	
haltsys		o
hd		o
hdinit		o
hdr		o
head		o
help	x	s
history		e
holidays	x	e
hp	x	
hpio	x	
hyphen	x	t
id	x	o
init	x	o
install	x	o
ipb	x	
ipcrm	x	o
ipcs	x	o
join	x	o
just		e
kasb	x	
kill	x	o
killall	x	
lastlogin	x	e
lc		o
ld	x	s
ld.pdp	x	
lex	x	s
line	x	o
link	x	o
lint	x	s
list	x	
ln	x	o
login	x	o
logname	x	o
look		o
lorder	x	s

Commands/ Applications	V	XENIX
lp	x	o
lpadmin	x	o
lpinit		o
lpr	x	o
lpmove	x	o
lpsched	x	o
lpshut	x	o
lpstat	x	o
ls	x	o
m4	x	s
machid	x	
mail	x	o
make	x	s
makekey	x	
man	x	t
masm		s
mesg	x	o
mkalias		e
mkboot	x	
mkdir	x	o
mkfs	x	o
mknod	x	o
mksalias		e
mkssites		e
mkstr		s
mkuser		o
mm	x	t
mmcheck		t
mmt	x	t
monacct	x	e
more		o
mount	x	o
msi	x	
mv	x	o
mvdir	x	o
ncheck	x	o
neqn	x	t
net	x	
netutil		o
newboot	x	
newfile		e
newform	x	o
newgrp	x	o
news	x	o
nice	x	o
nl	x	o
nm	x	s
nm.pdp	x	
nohup	x	o
nroff	x	t
nscloop	x	
nscmon	x	
nscstat	x	
nsctorje	x	
nulladm	x	e
nusend	x	
od	x	o
pack	x	o
passwd	x	o
paste	x	t
pcat	x	o
pcixread		e
pcixrestore		e
pcixls		e
pcldaemon	x	
pg	x	o
pr	x	o
prctmp	x	e

Commands/ Applications	V	XENIX
prdaily	x	e
prep		t
print		e
prm	x	
prof	x	s
profiler	x	
prs	x	s
prtacct	x	e
ps	x	o
pstat		o
ptx	x	t
pwadmin		o
pwcheck		o
pwck	x	o
pwd	x	o
qdaemon		e
qdisable		e
qenable		e
qftp		e
qhold		e
qstat		e
quot		o
random		o
ranlib		s
ratfor	x	s
rcp		o
readfile		e
readmill		e
reboot	x	o
red	x	o
regcmp	x	s
reject		o
remind		e
remote		o
remove		e
restor		o
restore		o
rfptsrvr		e
rjestat	x	
rm	x	o
rmail	x	o
rmdel	x	s
rmdir	x	o
rmhist		e
rmtcp		e
rmuser		o
rmv	x	
rpl		e
rsh	x	o
rst	x	
runacct	x	e
runbig		o
sact	x	s
sadp	x	
sag	x	
sar	x	
scat	x	
scc	x	
sccsdiff	x	s
sdb	x	
sddate		o
sdiff	x	s
se	x	
sed	x	o
send	x	
sendmail		e
setclock		o
setkey	x	o

Commands/ Applications	V	XENIX
setmnt	x	o
setmrf	x	
settime		o
sh	x	o
shutacct	x	e
shutdown	x	o
size	x	s
size.pdp	x	
sleep	x	o
sno	x	
soelim		t
sort	x	o
spell	x	t
spline	x	s
split	x	o
ssr	x	
st	x	
sta	x	
stackuse		s
startup		e
stat	x	
stgetty	x	
stlogin	x	
strings		s
strip	x	s
strip.pdp	x	
ststat	x	
stty	x	o
style		t
su	x	o
sum	x	o
sync	x	o
sysadmin		o
sysdef	x	
tabs	x	
tail	x	o
tar	x	o
tbl	x	t
tc	x	
tee	x	o
test	x	o
time	x	s
timex	x	
to		e
toc	x	
touch	x	o
tplot	x	
tr	x	o
troff	x	t
trouble	x	
true	x	o
tset		o
tsort	x	o
tty	x	o
turnacct	x	e
umask	x	o
umount	x	o
uname	x	o
unget	x	s
uniq	x	o
units	x	o
unlink	x	o
unpack	x	o
update	x	
uuclean	x	o
uucp	x	o
uuinstall		o
uulog	x	o

Commands/ Applications	V	XENIX
uumail		e
uuname	x	o
uupick	x	o
uustat	x	o
uusub	x	o
uuto	x	o
uux	x	o
val	x	s
vcf	x	
vc	x	
versions		e
vedit	x	o
vi	x	o
view	x	o
vlx	x	
volcopy	x	
vpmc.dec	x	
vpmc.u3b	x	
vpmsave	x	
vpmset	x	
vpmtest	x	
vpr	x	
vsh		o
wait	x	o
wall	x	o
wc	x	o
what	x	o
who	x	o
whodo	x	o
write	x	o
wtmp	x	
wtmpfix	x	e
x25pvc	x	
xargs	x	o
xconf		s
xcvt		s
xinstall		o
xld		s
xref		s
xstr		s
yacc	x	s
yes		s

System Calls	V	XENIX
access	x	s
acct	x	s
alarm	x	s
brk	x	s
chdir	x	s
chmod	x	s
chown	x	s
chroot	x	s
chsize		s
close	x	s
creat	x	s
creatsem		s
dup	x	s
dup2		s
execl	x	s
execle	x	s
execlp	x	s
execv	x	s
execve	x	s
execvp	x	s

System Calls	V	XENIX
exit	x	s
fcntl	x	s
fork	x	s
fstat	x	s
ftime		s
getegid	x	s
geteuid	x	s
getgid	x	s
getpgrp	x	s
getpid	x	s
getppid	x	s
getuid	x	s
ioctl	x	s
kill	x	s
link	x	s
lock		s
lockf	x	s
locking		s
lseek	x	s
maus	x	
mknod	x	s
mount	x	s
msgctl	x	s
msgget	x	s
msgrcv	x	s
msgsnd	x	s
nap		s
nice	x	s
open	x	s
opensem		s
pause	x	s
pipe	x	s
plock	x	s
proctl		s
profil	x	s
ptrace	x	s
putenv	x	s
rdchk		s
read	x	s
sbrk	x	s
sdenter		s
sdfree		s
sdget		s
sdgetv		s
sdleave		s
sdwaitv		s
semctl	x	s
semget	x	s
semop	x	s
setbuf	x	s
setgid	x	s
setpgrp	x	s
setuid	x	s
shmctl	x	s
shmget	x	s
shmat	x	s
shmdt	x	s
shutdn		s
signal	x	s
sigsem		s
stat	x	s
stime	x	s
sync	x	s
sys3b	x	
time	x	s
times	x	s
ulimit	x	s
umask	x	s

System Calls	V	XENIX
umount	x	s
uname	x	s
unlink	x	s
ustat	x	s
utime	x	s
wait	x	s
waitsem		s
write	x	s

Library Subroutines	V	XENIX
a641	x	s
abort	x	s
abs	x	s
acos	x	s
aimag	x	
aint	x	
asctime	x	s
asin	x	s
assert	x	s
atan	x	s
atan2	x	s
atof	x	s
atoi	x	s
atol	x	s
bool	x	
bsearch	x	s
burst_page		e
cabs	x	s
calloc	x	s
ceil	x	s
clearerr	x	s
clock	x	s
closedir		s
compile	x	s
conjg	x	
cos	x	s
cosh	x	s
crypt	x	
ctermid	x	s
ctime	x	s
curses	x	s
cuserid	x	s
dbminit		s
defopen		s
defread		s
delete		s
dial	x	
drand48	x	s
ecvt	x	s
end	x	s
endgrent	x	s
endpwent	x	s
endutent	x	s
erand48	x	s
erf	x	s
erfc	x	s
exp	x	s
fabs	x	s
fclose	x	s
fcvt	x	s
fdopen	x	s
feof	x	s
ferror	x	s

Library Subroutines	V	XENIX
fetch		s
fflush	x	s
fgetc	x	s
fgets	x	s
fileno	x	s
firstkey		s
floor	x	s
fmod	x	s
fopen	x	s
fprintf	x	s
fputc	x	s
fputs	x	s
fread	x	s
free	x	s
freopen	x	s
frexp	x	s
fscanf	x	s
fseek	x	s
ftell	x	s
ftok	x	s
ftw	x	s
ftype	x	
fwrite	x	s
gamma	x	s
gcvt	x	s
getarg	x	
getc	x	s
getchar	x	s
getcwd	x	s
getenv	x	s
getgid	x	s
getgrent	x	s
getgrgid	x	s
getgrnam	x	s
getlogin	x	s
getopt	x	s
getpass	x	s
getpw	x	s
getpwent	x	s
getpwnam	x	s
getpwuid	x	s
gets	x	s
getutent	x	s
getutid	x	s
getutline	x	s
getw	x	s
get_align		e
get_copies		e
get_endgroup		e
get_feed		e
get_from		e
get_header		e
get_lastuser		e
get_moddate		e
get_newuser		e
get_title		e
get_to		e
get_trailer		e
get_qdate		e
gmtime	x	s
gsignal	x	s
hcreate	x	s
hdestroy	x	s
hsearch	x	s
hypot	x	s
index	x	
isalnum	x	s
isalpha	x	s

Library Subroutines	V	XENIX
isascii	x	s
isatty	x	s
iscntrl	x	s
isdigit	x	s
isgraph	x	s
islower	x	s
isprint	x	s
ispunct	x	s
isspace	x	s
isupper	x	s
isxdigit	x	s
j0	x	s
j1	x	s
jn	x	s
jrand48	x	s
l3tol	x	s
l64a	x	s
lcong48	x	s
ldahread	x	
ldclose	x	
ldexp	x	s
ldfhread	x	
ldlread	x	
ldlseek	x	
ldohseek	x	
ldopen	x	
ldrseek	x	
ldshread	x	
ldsseek	x	
ldtbindex	x	
ldtbread	x	
ldtbseek	x	
lfind	x	s
len	x	
localtime	x	s
log	x	s
log10	x	s
log_charge		e
log_init		e
log_message		e
log_progress		e
log_status		e
longjmp	x	s
logname	x	s
lrand48	x	s
lsearch	x	s
ltol3	x	s
malloc	x	s
matherr	x	s
max	x	
mclock	x	
memccpy	x	s
memchr	x	s
memcmp	x	s
memcpy	x	s
memset	x	s
min	x	
mktemp	x	s
mod	x	
modf	x	s
monitor	x	s
mrand48	x	s
nbwaitsem		s
nextkey		s
nlist	x	s
nrand48	x	s
opendir		s
pclose	x	s

Library Subroutines	V	XENIX
perror	x	s
plot	x	
popen	x	s
pow	x	s
printf	x	s
putc	x	s
putchar	x	s
putpwent	x	s
puts	x	s
pututline	x	s
putw	x	s
put_header		e
put_trailer		e
qsort	x	s
rand	x	s
readdir		s
realloc	x	s
regcmp	x	s
regex	x	s
rewind	x	s
rewinddir		s
round	x	
scanf	x	s
seed48	x	s
seekdir		s
setvbuf	x	s
setgrent	x	s
setjmp	x	s
setkey	x	
setpwent	x	s
setutent	x	s
sgetl	x	s
sign	x	
sin	x	s
sinh	x	s
sleep	x	s
sprintf	x	s
sputl	x	s
sqrt	x	s
srand	x	s
srand48	x	s
sscanf	x	s
ssignal	x	s
stdio	x	s
step	x	s
store		s
strcat	x	s
strchr	x	s
strcmp	x	s
strcpy	x	s
strcspn	x	s

Library Subroutines	V	XENIX
strdup		s
strlen	x	s
strncat	x	s
strncmp	x	s
strncpy	x	s
strpbrk	x	s
strrchr	x	s
strspn	x	s
strtod	x	s
strtok	x	s
strtol	x	s
swab	x	s
system	x	s
tan	x	s
tanh	x	s
tdelete	x	s
telldir		s
tempnam	x	s
tfind	x	s
tgetent	x	s
tgetflag	x	s
tgetnum	x	s
tgetstr	x	s
tgoto	x	s
tmpfile	x	s
tmpnam	x	s
toascii	x	s
tolower	x	s
toupper	x	s
tputs	x	s
tsearch	x	s
ttyname	x	s
ttyslot	x	s
twalk	x	s
tzset	x	s
undial	x	
ungetc	x	s
utmpname	x	s
vfprintf	x	s
vprintf	x	s
vsprintf	x	s
x25alnk	x	
x25clnk	x	
x25hlnk	x	
x25ipvc	x	
y0	x	s
y1	x	s
yn	x	s

Application Program Development

The PC XENIX Software Development system allows the user to design and develop application programs for execution on the PC XENIX system. Part of the support for this consists of several libraries of subroutines and system calls which may be used by an application. There are more than 300 individual subroutines, system calls and macros provided.

These libraries are designed to be compatible with the AT&T UNIX System V Interface. Included are all of the standard system calls, the newer file locking, message queue, shared memory and semaphore calls. The subroutine libraries include standard input and output functions, math functions, string manipulation functions, display management functions and many more. These give the application writer a versatile set of predefined functions that should increase productivity by reducing the need to develop basic utilities. The following sections describe some of PC XENIX Software Development system's support in more detail.

Allocating Space

Some programs require significant changes to the size of their allocated memory space during different phases of their execution. The memory allocation functions of the standard C library let programs allocate space dynamically. This means that a program can request a given number of bytes of storage for its exclusive use at the moment it needs the space, then free this space after it has finished using it.

There are four memory allocation functions: malloc, calloc, realloc and free. Use the "malloc" and "calloc" functions to allocate space for the first time. The functions allocate a given number of bytes and return a pointer to the new space. The "realloc" function reallocates an existing space, allowing one to use it in a different way. The "free" function returns allocated space to the system.

The malloc function allocates space for a variable. The function call has the following form:

```
malloc(size)
```

where size is an unsigned number that gives the number of bytes you want to allocate. For example, the function call:

```
table = malloc(4);
```

allocates 4 bytes of storage. The function returns a pointer to the starting address of the allocated space. It returns the null pointer value if there is not enough space to allocate. The malloc function can also allocate storage for a group of strings that vary in length.

The calloc function allocates storage for a given array and initializes each element in a new array to zero. The function call has the following form:

```
calloc(n, size)
```

where n is the number of elements in the array, and size is the number of bytes in each element.

The function normally returns a pointer to the starting address of the allocated space. It returns a null pointer value if there is not enough memory. For example, the function call:

```
table = calloc(10, 4);
```

allocates space for a 10-element array. Each element occupies 4 bytes.

The realloc function reallocates the space at a given address without changing the contents of the memory space. The function call has the following form:

```
realloc(ptr, size)
```

where ptr is a pointer to the starting address of the space you want to reallocate, and size is an unsigned number giving the new size in bytes of the reallocated space. The function normally returns a pointer to the starting address of the allocated space. It returns a null pointer value if there is not enough space to reallocate the storage.

The free function frees unused memory space that had been previously allocated by a malloc, calloc, or realloc function call. The function call has the following form:

```
free(ptr)
```

where ptr is the pointer to the starting address of the space you want to free. This pointer must be the return value of a malloc, calloc, or realloc function.

File Locking

Locking a file is a way to synchronize file use when several processes may require access to a single file. The standard C library provides this file locking function, "lockf". This function locks any given section of a file, preventing all other processes that wish to use the section from gaining access. A process may lock the entire file or only a portion of

it. In any case, only the locked section is protected; all other sections can be accessed by other processes as usual.

Before you can lock a file, you must first open it using the open function, and properly position it with the "lseek" function to move the file's character pointer to the first byte you want to lock. Use the open function once at the beginning of the program to open the file. Use the lseek function any number of times to move the character pointer to each new section you want to lock. For example, the following statements locate the file pointer at the byte position 1024 from the beginning of the file named reservations for locking:

```
fd = open("reservations",O_RDONLY);

lseek (fd, 1024, 0);
```

The lockf function locks 1 or more bytes of a given file. The function call has the following form:

```
lockf(fildes, function, size)
```

where fildes is the file descriptor of the file locked, function is an integer value that defines the type of lock applied to the file. The size is a long integer value giving the size in bytes of the portion of the file section locked or unlocked.

File locking provides unhindered access to any portion of a file for a controlling process, and it protects a file from the damage caused by several processes trying to read or write to the file at the same time. To use the lockf function to lock a file, you must add the following line to the beginning of the program:

```
#include <unistd.h>
```

The file unistd.h contains definitions for the modes used with the function.

Semaphore Calls

Semaphores provide a general method of communication between two processes that are an extension of the features of signals. Semaphores are used in much the same way as signals, except that semaphores are:

- More flexible: Processes can define a semaphore to mean what they want it to mean.
- More controllable: Programs have direct control over semaphores and do not need to depend on the system to generate them.
- Broader in scope: A semaphore can be any integer value, not just 1 or 0.

The system calls that allow a program to use semaphores are:

Call	Description
semctl()	Semaphore control operations
semget()	Get set of semaphores
semop()	Semaphore operations

Semaphores are counters that a program can test and change with a single system call, semop(). Use semaphores for passing data between processes and for other one time data transfers. Semaphores can also be used to control access to a limited resource, such as a shared buffer.

Message Calls

Messages provide a general method of communication between two processes. Using messages, one process can pass information to another process. The information may be data that is produced by one process and used in another process, or it could be flags that indicate when events occur. To use the message process, perform the following steps:

1. Use the ftok() subroutine to get a key assigned to a message queue.
2. Use the msgget() call to get a message queue assigned to the processes.
3. Use the msgsnd() call to send a message to a queue that is assigned to another process.
4. Use the msgrcv() call to receive a message from the message queue.

Use the following system calls to create and use message queues:

Call	Description
msgctl()	Gets status, change permissions, or removes a queue
msgget()	Gets message queue
msgrcv()	Receives a message
msgsnd()	Sends a message

Include the following files in your program when using message queue calls:

```
#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/msg.h>
```

In effect, message queues are a more general form of the pipe system call. Either method passes information between two processes. However, for message queues you do not need to perform the steps of opening a pipe, forking and then closing two ends of the pipe. Also, the two processes using message queues to communicate do not need to be created from the same ancestor process; they only need to cooperate by using the same name for the queue and agreeing about what the messages mean.

Shared Memory

The shared memory calls set aside an area of memory that other processes can access. This area serves as a large pool for exchanging data among the processes. The shared memory calls do not provide locks or access control among the processes. Therefore, processes using the shared memory area must set up a signal or semaphore to prevent access conflicts and to keep one process from changing data that another process is using. Use shared memory when the amount of data is too large to transfer with messages, or when many processes maintain a common large data base.

Use the following calls to create and use shared memory segments from a program.

Call	Description
shmctl()	Controls shared memory operations
shmget()	Gets or creates a shared memory segment
shmatv()	Attaches a shared memory segment to a process
shmdt()	Detaches a shared memory segment from a process

Include the following files in your program when using shared memory calls:

```
#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/shm.h>
```

Standard I/O Functions

IBM PC XENIX provides the following standard I/O functions in the standard C library.

- Command line arguments
- Standard input and output files
- Stream functions for ordinary files
- Low-level functions for ordinary files
- Random access functions

To use the standard I/O functions, a program must include the file "stdio.h". This file defines the macros and variables needed to use the standard I/O functions. To include this file, place the following line at the beginning of your program:

```
#include <stdio.h>
```

The library file "libc.a" contains the functions. The compiler reads this file whenever you compile a program.

The standard I/O library uses many names for special purposes. You can use these names in any program that has included the stdio.h file. The special names are:

stdin	The name of the standard input file.
stdout	The name of the standard output file.
stderr	The name of the standard error file.
EOF	The value returned by the read routines on end-of-file or error.
NULL	The null pointer, returned by pointer-valued functions, to indicate an error.
FILE	The name of the file type used to declare pointers to streams.
BUFSIZ	The size in bytes suitable for an I/O buffer supplied by the user.

Whenever you invoke a program for execution, PC XENIX creates a standard input, a standard output and a standard error file to handle the input and output needs of the program. The bulk of input and output of most programs is through your own terminal. The system, therefore, assigns your terminal keyboard and screen as the standard input and output respectively. It also assigns the

standard error file, which receives any error messages generated by the program, to your screen.

A program can read and write to the standard input and output files with the getchar, gets, scanf, putchar, puts and printf functions.

PC XENIX allows you to redirect the standard input and output using the shell's redirection symbols (< and >). This allows a program to use other devices and files as its chief source of input and output.

String Manipulation Functions

The PC XENIX string functions can concatenate, compare, copy and count the characters in a string. Two special string functions, sscanf and skrintf, let a program read from and write to a string in the same way the standard input and output can be read and written. These functions are convenient when needing to read or write whole lines containing values of several formats. It is not necessary to use any command to use the string functions. The standard C library defines the functions and they are read whenever one compiles a C program.

Many string functions have two forms: a form that manipulates all characters in the string and one that manipulates a given number of characters. This gives programs control over all or part of strings. Described in more detail below are some of PC XENIX's string manipulation functions.

The strcat function concatenates two strings by appending the characters of one string to the end of another. The function call has the following form:

```
strcat(dst, src)
```

where dst is a pointer to the string to receive the new characters, and src is a pointer to the string containing the new characters. The function appends the new characters in the same order as they appear in src, then appends a null character (\0) to the last character in the new string. Make the receiving string large enough to contain the new string. The function cannot detect overflow. The function always returns the pointer dst.

The strcpy function copies a given string to a given location. The function call has the following form:

```
strcpy (dst, src)
```

Where dst is a pointer to the location to receive the string, and src is a pointer to the string strcpy copies. The function copies all characters in the source string src to dst and appends a null character (\0) to the end of the new string. The function returns the pointer to the new string.

The strlen function returns the number of characters in a given string. The function call has the following form:

```
strlen(s)
```

where s is a pointer to a string. The count includes all characters up to, but not including, the first null character. The return value is always an integer.

Math Functions

PC XENIX provides a standard math library containing many valuable mathematical functions. Math functions include:

exp	Exp performs the exponential function of x.
log	Log performs the natural logarithm of x.
sqrt	Sqrt performs the square root of x.
sin,cosh, tanh	These functions compute the hyperbolic functions for real arguments.

There is an error-handling function, matherr, that is invoked by functions in the Math Library when errors are detected. Users may define their own procedures for handling errors by including a function named matherr in their programs. When an error occurs, a pointer to the exception structure x will be passed to the user-supplied matherr function. If matherr is not supplied by the user, the default error-handling procedures, described with the math functions involved, will be invoked upon error.

Utilities and Commands

The PC XENIX Software Development system provides the user with many utilities to help design and develop application programs. These utilities help the user to create C language or assembler language programs for execution on the PC XENIX system. Utilities also are provided to allow the user to automate program creation, interactively debug programs and maintain different versions of the same program.

You can effectively create C language source files using a text editor provided with the PC XENIX

Operating System. The vi editor is a full-screen editor which provides many commands to insert, replace, move and search for text.

Once the source program is created, it can be compiled with the C language compiler. The "cc" command invokes the compiler. The cc command is used to invoke the C compiler and other utilities such as the link editor "ld" and the assembler "as". The ld command is used to invoke the link editor separately, but ld should be used only with the PC XENIX C compiler.

One can debug an executable C program with the debugger "adb". The adb debugger provides a direct interface to the machine instructions that make up an executable program.

To check a program before compiling it, use the C program checker "lint". The lint utility program checks for syntactical and logical errors. It also enforces a strict set of guidelines for C language programming style. The lint utility is normally used in the early stages of program development.

To improve a program's format, use the C program beautifier "cb". The beautifier improves the appearance of C language programs and is designed to make them easier to read.

The C language can meet the needs of most programming projects but, in some cases, you may want to create assembler languages programs where greater control might be required. Assembler language programs are created using the "as" program. This program assembles source files and produces object files. One can then relocate or link the object files to C language programs.

Libraries of functions and programs can be created and maintained. The archiver utility "ar" creates libraries of relocatable object files. The "ranlib" utility converts archive libraries to random libraries and places a table of contents at the beginning of each library. The "lorder" command finds the ordering relationship in an object library and produces a list of dependent pairs. The "tsort" command sorts the dependent pairs into an order that shows their dependencies.

The make utility is a program source file maintainer. It automates the steps required to create executable programs and provides a mechanism for ensuring up-to-date programs. You should generally use make with large-scale programming projects.

The Source Code Control System (SCCS) is a collection of commands that create, maintain and control special files called SCCS files. The SCCS commands let you maintain different versions of a single program. This is done by storing the original program and each set of changes. The commands compress all versions of a source file into a single file containing a list of differences. These commands can also restore compressed files to their original size and content.

The PC XENIX Software Development system contains many application development tools to provide an efficient software application development environment for the design and development of application programs.

A Portability Consideration

One advantage of IBM Personal Computer XENIX is that, since it is based on an AT&T UNIX System V, applications developed for PC XENIX will have a high degree of portability to similar operating systems, such as IBM Advanced Interactive Executive for the IBM RT Personal Computer. It should be noted, however, that some of these systems run on 32-bit processors, while others, like PC XENIX, run on 16-bit processors. This is nowhere more evident than in the matter of the process identifiers (PIDs) used by these systems to reference individual tasks. Applications which assume that PIDs are 16 bits long limit their portability to 32-bit systems. Developers are encouraged, therefore, to store PIDs in "long integers" (32 bits) rather than in "short integers" (16 bits) whenever possible. This will reduce problems when moving applications between different processor types.

Consistent Installation Method

One way that the user friendliness of PC XENIX has been improved is by the provision of a common mechanism for installing all software for the system. The "xinstall" program was used in PC XENIX version 1.00 to install the Operating System, Software Development System and the Text Formatting System program products. Xinstall has been improved in Version 2.00 to allow the user to install any application or system enhancement package that meets certain criteria.

The criteria that a package must meet are as follows:

1. The package must consist of 5 1/4 inch, 96TPI, "TAR" format diskettes. No files should cross a volume boundary.

2. The package should contain an executable file called /etc/extninit. This file should be used to perform any necessary initialization or configuration for a package. It will be run by xinstall, then removed.

3. The package should contain an ASCII file called /etc/extnname. This file should contain the name of the package which will be displayed by xinstall. The file will then be removed.

4. The package should contain an ASCII file called /etc/extnbook. This file should contain the name of the reference manual the user should be using during installation. This name will be displayed by xinstall, then removed.

When a software package meets these criteria, the user should be directed by the installation section of the package's documentation to type "xinstall extn", and follow the instructions on the screen. This allows users the advantage of using a familiar mechanism when installing additional software.

Application Compatibility

The major concern of most users and developers is that of application compatibility. Simply stated, "What software do I now have that will execute on this machine or operating system?" This section is intended to provide answers to this question and others that arise from it.

In order to more accurately address the application compatibility issue, it is necessary to discuss both execution level compatibility and source level compatibility.

PC XENIX Applications

The IBM Personal Computer XENIX Operating System Version 2.00 is binary-compatible with programs developed on IBM Personal Computer XENIX Operating System Version 1.00. Programs developed for Version 1.00 will execute on Version 2.00. Programs written according to the UNIX System V Interface Definitions (issue Number 1, dated Spring 1985) are source-code compatible, provided they use only the functions supported by the IBM Personal Computer XENIX Operating System Version 2.00.

Commands not supported are:

```
bs          sno
cflow       compress
conv        crypt
convert     ct
cxref       mailx
ctc         makekey
ctcr        shl
ctrace      tabs
list        uusched
sdb
```

Libraries not supported are:

```
libg        libmalloc
libld       libPW
```

Subroutines not supported are:

```
crypt
encrypt
setkey
```

UNIX System V Applications

PC XENIX is derived form the UNIX System V Operating System. This means that the majority of the kernel, utilities and libraries were migrated directly from the distributed AT&T source code.

The ability to migrate UNIX System V based source code to PC XENIX is possible. But, a number of problems arise if the source code attempts to utilize any of the following C language features:

- Integer/Pointer Coercion
- Pointer Reference to Control Hardware
- Pointer Reference to Control Software
- Machine Dependent Coding Techniques

If the programming techniques used in the application conform to the standard I/O library interface of AT&T UNIX System V, then the task of migrating the application is reduced.

The ability to migrate applications (other than from IBM PC XENIX) at the object or executable level is virtually nonexistent. The major reasons are listed below:

- Different Processor Instruction Sets
- Different Executable File Formats
- System Call Initiation Differences

Therefore, the only level of compatibility which exists between applications running in another UNIX System V environment and PC XENIX is at the source level.

PC/IX Applications

The level of application compatibility between PC/IX applications and PC XENIX is essentially the same as that for standard UNIX System V applications. Migration will be more difficult if the PC/IX applications use:

- System III functions that are not supported in System V.
- PC/IX extensions that are not supported in PC XENIX.

PC DOS Applications

Migrating PC DOS applications source code to PC XENIX is not an easy task. There are a number of problems which exist. The major problem areas are caused by the different operating environments of the two products. For example, PC DOS applications have the following characteristics:

- Nonreentrant Code
- Fixed Addressing After Initial Load
- Nonswappable Code
- Total Machine Resource Philosophy

On the other hand, UNIX applications conform to a much different set of guidelines and have the following different characteristics:

- Reentrant Code
- Swappable Code
- Dynamic Relocation
- Process/Kernel Resource Philosophy

Many C language applications can be migrated from the PC DOS environment to the PC XENIX environment if the following features are avoided:

- Integer/Pointer Coercion
- Pointer Reference to Control Hardware
- Pointer Reference to Control Software
- Machine Dependent Coding Techniques

Likewise, if the programming techniques used in the application are relatively clean and conform to the basic standard I/O library interface, then the task of migrating the application is greatly reduced.

Summary

IBM PC XENIX is a full function, multitasking, multiuser operating system which exploits the capabilities of the advanced Intel 80286 architecture. The PC XENIX family of products brings high function to most environments. It contains features useful in the software application development environment, and it has tailored function for those environments producing technical and professional documents. It also contains additional functions designed to enhance compatibility between PC XENIX and other UNIX derivative products. The PC XENIX system brings many of the features normally found on mainframe computers to the IBM PC AT.

IBM Personal Computer Seminar Proceedings

Publication Number	Volume	Topic
	V1.1	*Contains identical information as V1.2*
(G320-9307)	V1.2	IBM PC DOS 2.0 and 1.1 Comparison Compatibility Guidelines for Application Development 8087 Math Co-Processor IBM Macro Assembler
(G320-9308)	V1.3	IBM PC DOS 2.1 & Comparison to DOS 2.0 and 1.1 IBM PC*jr* Architecture & Compatibility Overview Cartridge BASIC IBM Personal Communications Manager-Modem Drivers
(G320-9309)	V2.1	*Contains identical information as V2.2*
(G320-9310)	V2.2	IBM Software Support Center International Compatibility Requirements IBM Personal Computer Cluster Program
(G320-9311)	V2.3	IBM Personal Computer Cluster Program Sort, Version 1.00 FORTRAN and Pascal Compiler, Version 2.00 PC*jr* Cartridge Tips and Techniques
(G320-9312)	V2.4	IBM Personal Computer AT Architecture ROM BIOS Compatibility & Software Compatibility IBM PC DOS 3.0
(G320-9313)	V2.5	IBM PC Network Overview, Hardware & Program IBM PC Network BIOS (NETBIOS) Architecture
(G320-9314)	V2.6-1	TopView
(G320-9315)	V2.7	IBM Personal Computer Resident Debug Tool
(G320-9319)	V2.8-1	IBM PC Network SMB Protocol
(G320-9316)	V2.9	IBM Personal Computer XENIX, Version 1.00
(G320-9317)	V2.10	IBM PC Professional Graphics Software IBM PC Graphical Kernel & File Systems IBM Plotting System Library IBM Professional FORTRAN IBM PC Data Acquisition & Control Adapter & Software IBM General Purpose Interface Bus Adapter & Software
(G320-9318)	V2.11-1	IBM Enhanced Graphics Adapter
(G320-9320)	V3.1	IBM PC Information Panel (3295 Plasma Display)
(G320-9321)	V3.2	IBM BASIC Compiler 2.00
(G320-9322)	V3.3	IBM Personal Computer C Compiler
(G320-9323)	V3.4	IBM Asynchronous Communications Server Protocol
(G320-9324)	V3.5	IBM Personal Computer Voice Communications Option
(G320-9325)	V4.1	IBM Personal Computer XENIX, Version 2.00

G320-9325

IBM Corporation
Editor, IBM Personal Computer Seminar Proceedings
Internal Zip 4629
Post Office Box 1328
Boca Raton, FL 33429-1328