# IBM PC/IX

## Standard UNIX runs better than you think for the PC

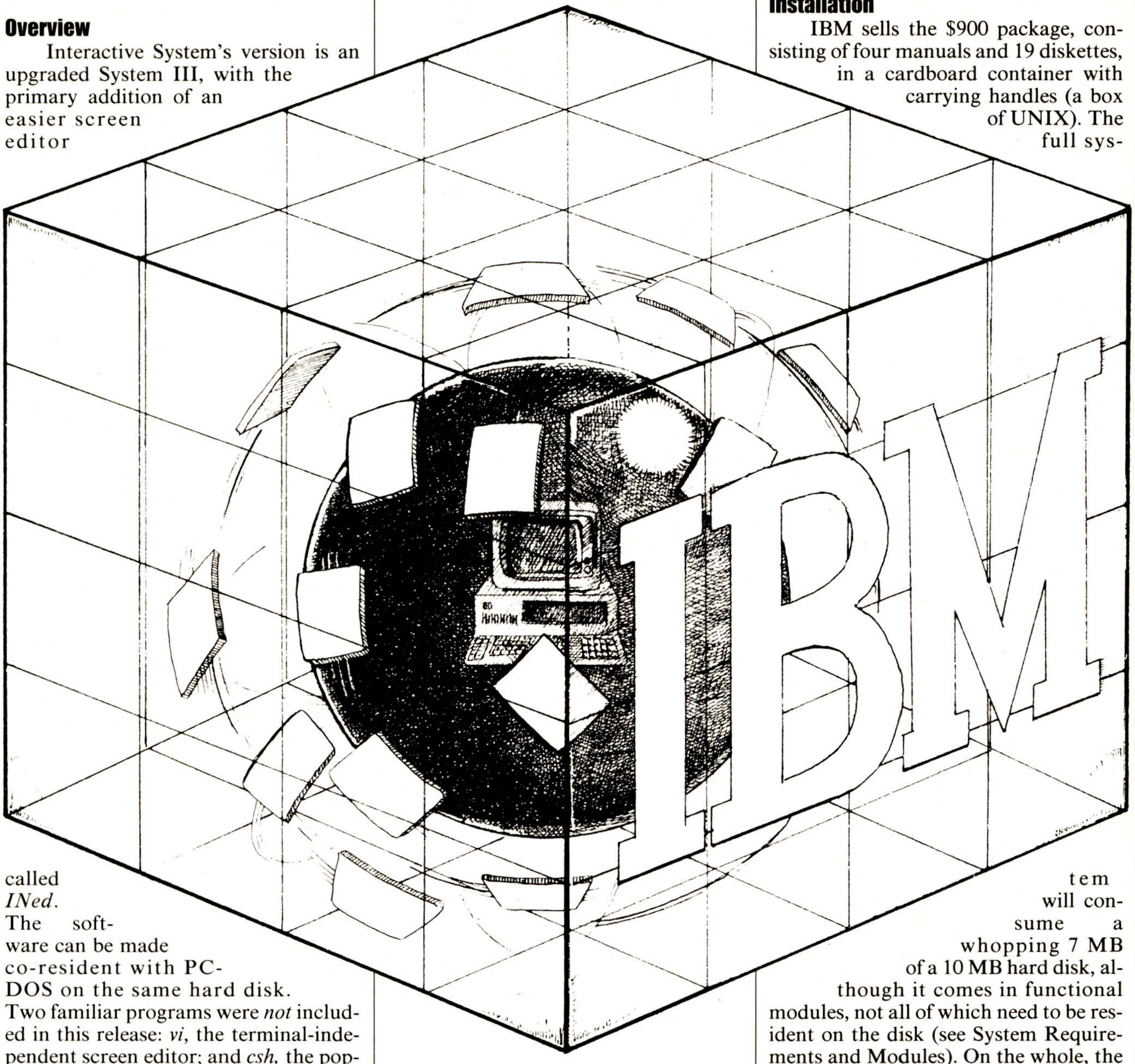
by Peter Brooks

The battle of the titans has spread to our backyard. AT&T, formerly the world's largest private company in terms of employees, and having arguably the world's best research and development arm (Bell Laboratories), announced an IBM PC-compatible micro in June. In the same month IBM, the world's most profitable private company, delivered PC/IX, a UNIX system for its PC and PC XT. This is the first skirmish in microcomputers by the giants, with more to come soon.

For those needing more software power than PC-DOS delivers, the new crop of UNIX offerings provides the tools to build large and complex systems. All the traditional strengths of UNIX—machine independence, timesharing, modular design, abundance of simple tools and an 'open' software architecture—have now been brought down from the minis to serve microcomputer users and software developers. And the late 1970's generation of university programmers who grew up under a UNIX environment because Western Electric used to give it away for practically nothing, have now entered the industry and brought their love for UNIX with them. They continue to communicate over an active network of UNIX electronic mail system connections that carpet the country.

PC/IX (Personal Computer Interactive Executive—some people will go to great lengths to obfuscate an obvious acronym) was developed by Interactive Systems Corporation to be marketed exclusively by IBM as a single-user version of UNIX (see sidebar). This is a solid product, with all of the familiar software development tools that UNIX programmers have treasured over the years.

Even without a direct AT&T offering, the competition in micro versions of UNIX has already heated up. Based on Bell Labs' UNIX System III, PC/IX will be in head-to-head competition with XENIX from MicroSoft (which is also based on System III), VENIX/86 from VentureCom (based on Version 7), and a variety of UNIX look-alikes for the PC (QNX from Quantum Software, Coherent from Mark Williams, IDRIS from Whitesmiths). While UNIX System V is the most advanced version of the operating system on the market, and heavily marketed by AT&T, IBM presumably chose System III because it had been out in the field for years and had been extensively tested.



### Overview

Interactive System's version is an upgraded System III, with the primary addition of an easier screen editor called *INed*. The software can be made co-resident with PC-DOS on the same hard disk. Two familiar programs were *not* included in this release: *vi*, the terminal-independent screen editor; and *csh*, the popular C-shell. Rumor has it that IBM is expected to release these products in a future version, but for good reason did not believe that *vi* was necessary in a single-user UNIX environment alongside *INed*. The omission of *csh* was probably for reasons of space.

One of the reasons for the great success of the PC was the decision to publish the internal hardware specifications and the contents of the ROM BIOS. This generated an entire industry of hardware and software developers who knew virtually everything they needed to know about the machine. With PC/IX, IBM has again implemented their 'open architecture' design philosophy. They provide both the required software and tutorial help for implementing new device drivers.

**The traditional strengths of UNIX have been brought from minis to micros.**

### Installation

IBM sells the $900 package, consisting of four manuals and 19 diskettes, in a cardboard container with carrying handles (a box of UNIX). The full system will consume a whopping 7 MB of a 10 MB hard disk, although it comes in functional modules, not all of which need to be resident on the disk (see System Requirements and Modules). On the whole, the documentation is better than for any other UNIX system I've seen, with very good tutorials, articles, and modification suggestions.

The most important decision in installation is: how much of the hard disk are you willing to allocate to PC/IX, and how much to PC-DOS? The two systems can be made co-resident only by allocating fixed areas of the disk to each. Therefore, if you had installed PC-DOS without the expectation of using the disk for anything else, you will first have to back up all of your files onto floppies, split the disk into partitions of predetermined size (erasing all information from the hard disk), and reload your PC-DOS files into the PC-DOS partition. After installing both systems, you can

Continued from page 67

switch between the two very easily, although only one system can be running at any given time.

An area of 6-8 MB is recommended for PC/IX, depending upon which of the optional software modules you expect to use and how much space you need left over for your own files. You can, of course, allocate the entire disk to PC/IX and run PC-DOS from floppies, or alternatively purchase another 10 MB hard disk (which should speed up PC/IX multitasking operations significantly by locating the swap area on a different disk).

Once the disk has been partitioned, there is a 24-step installation procedure to load the core system that takes about half an hour. Then, one can load any or all of the optional modules (see box on Software Modules). It takes about 2 hours to accomplish a full system startup for someone who has done it before, and not much longer for the rest of us who like to know what we're doing along the way.

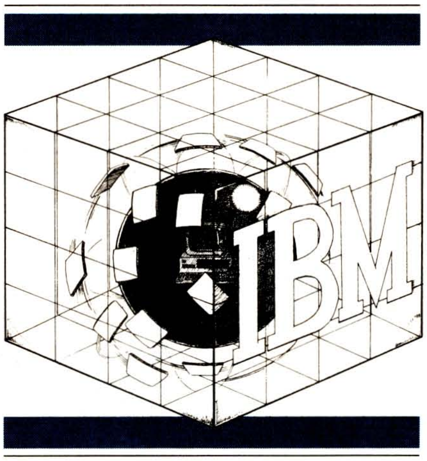
Although the documentation doesn't mention how, the license agreement allows the creation of backup diskettes. It turns out that PC-DOS' *diskcopy* works nicely.

Since the programs are designed to run on a single 10 MB disk, and the full 19-diskette system swallows at least 6.4 MB, IBM has carefully broken the programs apart into subsets. The *core*, consisting of all of the commonly used and necessary commands plus the editor, comprises the minimum usable subset. It takes up 1/3 of the disk (3.4 MB), not including user space, and allows the system to perform virtually any small task. The rest of the subsets are optional, and can be loaded at any time.

Of course, any serious development work would probably require at least the C compiler, and its associated utilities like *lint*, the C program checker, *adb*, the debugger, and the function libraries, all of which are contained in the programming subset. This is the largest of the modules, taking up 1.3 MB on four diskettes.

The communications subset contains the powerful UNIX file transfer and terminal programs, and the unix-to-unix mail facilities. The source code control system module consists of programs to track versions of files that change often. With the text processing subset, the user receives the UNIX text formatting languages and filters: *nroff, tbl, mm*, etc. In addition, the special-purpose module provides the programs for actually typesetting materials. With system accounting installed, the operating system will keep track of logins, disk, printer and CPU usage, and produce summary reports on a periodic basis.

The final subset is the games, presumably intended to keep the programmer busy while the machine is doing real work. While the chess program (rated class D player) is not included, tic-tac-toe, backgammon and blackjack might keep the user amused, along with a multitude of quiz programs and fortune cookie sayings.



## Documentation

The four manuals included should serve as a complete guide to the system.

Beginners can start with the Text Processing Guide, which includes a tutorial on the file system and the use of the basic commands. Most valuable is the tutorial for *INed*, with step-by-step instructions on the creation and editing of simple documents and files. Besides the tutorials, and the INed reference guide, all other documents in this manual are reprints from the UNIX literature: guides to *ed*, the line editor; *nroff*, the text formatter; *troff*, the typesetting formatter; *mm*, the formatting macros for *nroff* and *troff*; and *tbl* and *eqn*, filters to create tables and mathematical equations in documents.

The System Manager's Guide is especially well done, with explanations of setting up ports for communication, doing system accounting, adding users, and figuring out the line printer spooler. Care and feeding of the file system is covered in detail. Various types of modems are already supported by the system (Hayes Smartmodem 1200, VenTel MD212-Plus and DEC DF03), and help is provided for setting the switches and creating autodialer programs. Since there is very little difference between the communications programs that IBM provides and the standard complement of UNIX networking tools, the manual includes reprints from the UNIX literature by the authors of these tools. And for the intrepid, there is a guide to writing device drivers and embedding them into the operating system.

The Programmer's Guide, a fat volume of 250 pages, contains chapters on all of the major programmer's tools: the shell; the C language; *lint*; the assembler (as difficult to use here as on other UNIX systems); the debugger; *make*; *sccs*; *lex*; and *yacc*; the macro processor, *m4*; the stream editor, *sed*; the pattern scanning language, *awk*; the terminal-independent screen library, *curses*; and the desk calculator languages, *bc* and *dc*. Virtually all of the material comes from UNIX reprints, usually from the authors of the tools themselves.

The User's Manual uses the same format as all other UNIX User's Manuals for describing the billions of commands available on the system. The explanations are terse and often difficult to figure out, even if you are used to the style. The format is at least honest, with a subsection of each command devoted to known bugs. Since there is no online version of the manual (as there is on most large UNIX systems) because of the shortage of disk space, the written version becomes especially important.

All told, the documentation is reasonably complete; nevertheless, I defy anyone to master system accounting without extensive experimentation (the following note appears at the beginning of the accounting section of the manual: "No manual can take the place of good, solid experience.") Despite the volume of print thrown at the purchaser, some important topics were left out. There are no indexes in any of the manuals (except the User's Manual), and no discussion on how to make backup diskettes from the originals. There is a crying need for some sort of map of the file system—where things are—since a fully loaded disk contains 700-800 files spread over 80 directories (Figure 1). Finally, there is no introductory discussion of what most of the tools do, or how they fit together. On the other hand, there is much more here than is normally provided to a UNIX user.

## Performance

As one would expect, a large machine system brought down to the microcomputers will suffer performance degradation. PC/IX is no exception. The PC's processor is slow, the hard disk is slow, and the software speed matches the hardware. As a single-user system, however, the performance is quite acceptable. Even with processes executing in the background—provided they are given low priorities—the user will not be left waiting.

Some of tools, like *awk*, the C compiler, and the text formatters, can drag the system to a crawl. Unfortunately, one of the worst offenders is Interactive System's editor, *INed*. It takes 10-15 seconds to invoke each time, even with a small file. Fortunately, because of the editor's design, one does not need to call it from scratch each time a file needs revision (see the discussion below). Most of the ordinary tools perform much faster.

With multiple users, the story is different. Each person knows the load they have placed on the system, and will compensate psychologically for degraded response if a large request has been invoked. However, if others also make requests, then the response is unpredictable, especially if it is slow. With such a small machine, each new task takes perceptible time away from other tasks, particularly interactive ones, and the user can feel the difference. Microcomputer users, spoiled by having entire machines dedicated to their whims, may be loath to give up such pleasures for the unknown benefits of timesharing.

## INed—the screen editor

*Ined* is the primary addition to this otherwise standard UNIX System III implementation, and appears to be a descendent of an older Interactive Systems editor called Ned. It's programmed to interact directly with the PC's video RAM, making visual updates much faster, but also making *INed* impossible to use by any device other than the console (see sidebar on single-userhood). The system administrator should ensure that anyone else logging in be prevented from using *INed*, because the console screen will go haywire!)

INed is easier to learn from scratch than *vi*, because many of the simple commands, such as cursor movement, are dedicated to appropriate special keys on the keyboard. Besides the tutorial and the reference manual, there is online help for each of the commands, and a small plastic map of the keyboard to be placed near it for reference.

All of the standard text editor functions are available: change tabs and margins, center, search and replace, move text blocks and columns, format with or without right justification. While most of the simple operations are fast, both scrolling and searching can be somewhat slow. The screen mimics a blank sheet of paper better than in *vi* or WordStar, and the cursor can be moved anywhere on it without regard to previous 'end of lines' and other hangups. The cost of this blank sheet approach is that it is as hard in *INed* as in *vi* to split a line or join two adjacent lines together.

**Comparison of major programs with UNIX System III**

| | |
|---|---|
| **Added** | |
| dosread | PC-DOS utilities |
| doswrite | |
| dosdir | |
| dosdel | |
| e | INed visual editor |
| format | Format diskettes |
| **Included** | |
| acct- | System & user accounting tools |
| adb | C program debugger |
| ar | Archive and library manager |
| as | 8088/6 assembler |
| awk | Text processing language |
| bc & dc | Desk calculator utilities |
| bs | Basic/Snobol interpreter/compiler |
| cc | C compiler |
| curses | Screen management library |
| ed | Line editor |
| ld | Link editor |
| lex | Lexical parser |
| lint | C program checker |
| m4 | Macro processor |
| mail | Send mail |
| make | Program module maintenance |
| mm | n/troff macro library |
| n/troff | Text formatters |
| prof | Program profiler |
| sccs | Source code control system |
| sed | Stream editor |
| sh | Bourne shell |
| sno | Snobol interpreter |
| sort | Sort and merge |
| stty | Set terminal characteristics |
| uucp | unix-unix copy |
| yacc | Compiler-compiler |
| **Omitted** | |
| csh | C-shell |
| f77 | Fortran 77 |
| man | (no manual pages online) |
| ms | n/troff macro library |
| tar | Tape archiver (no tape utilites) |
| vi | Visual editor |

*INed* supports multiple editing windows, cutting the screen either vertically or horizontally, and you can easily pick up text in one window and plant it in the other. You can sequentially edit files without leaving the editor.

A fancier feature is the ability to run programs (filters) on all or a portion of the file that you are currently editing. For example, you can request, while in the middle of a file, that the next 27 lines of text be sorted. *INed* will call the *sort* program and replace the next 27 lines on the screen with its output. Virtually any other filter will work the same way. This mechanism can be used to extract a portion of the current file (using *cat* as a filter), eliminate selected parts of it (using *grep*), even format it, using *nroff* (although there are quicker ways of doing simple formatting). As you become more familiar with the powerful UNIX text manipulation programs, you can watch them work dynamically on the text in front of your nose.

As with other UNIX editors, you can escape to the shell in the middle of editing a file. *INed* will save the file before creating a new shell, so it can be compiled or manipulated in other ways. Exiting the shell will pop you back into the file where you left it. Therefore, you never need to exit the editor. You can also ask the editor to run a program and put its output into a box on the editing screen (*popbox*). Thus, you can ask for the date without having to take your eyes off the text, or run a telephone program that will dial someone, given the person's name.

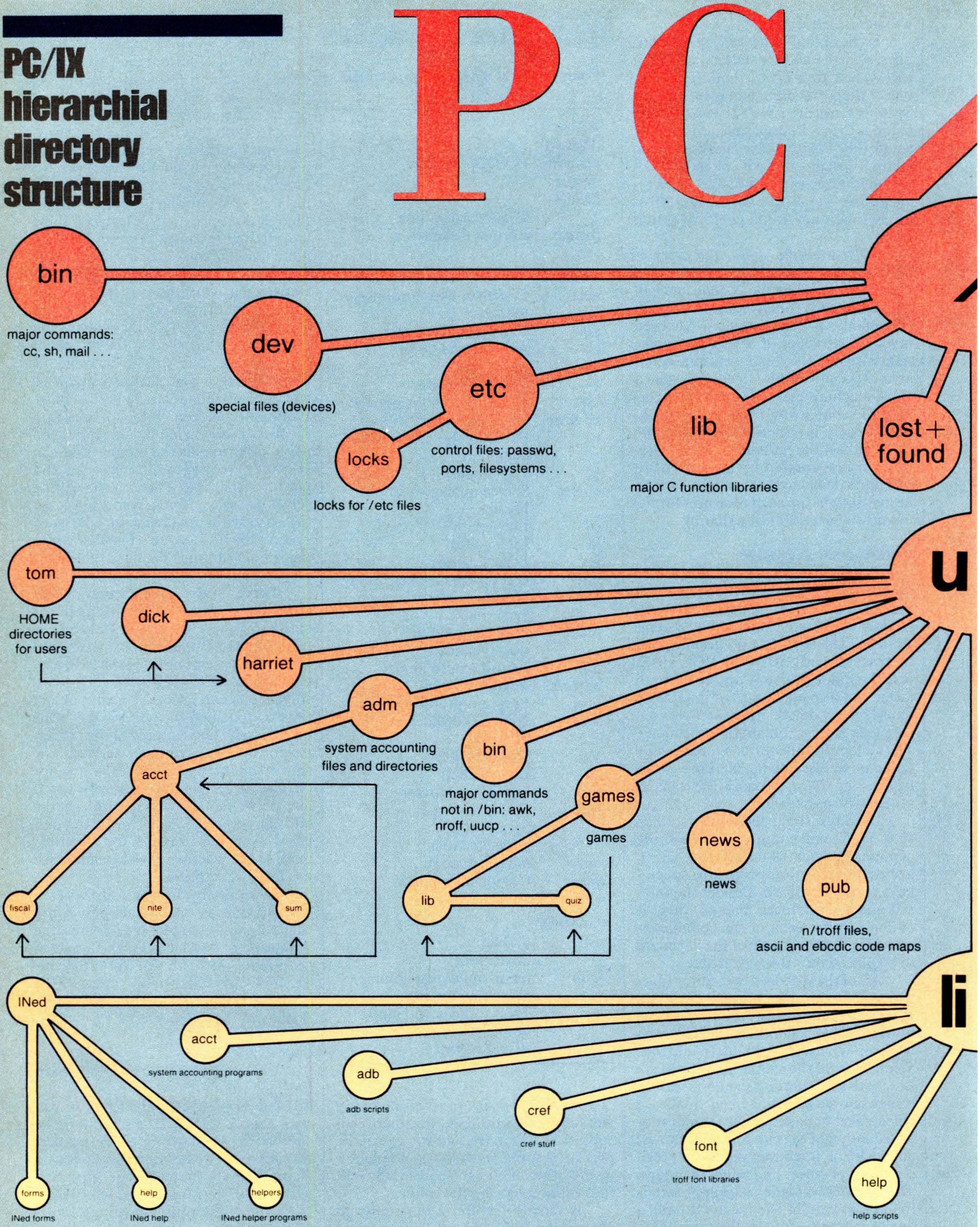
There is also a configuration file that will allow some degree of customization of the editor. Help menus can be changed to suit your needs, and active menu items inserted that will execute programs or shell scripts. Once installed, you can select the menus at any time by touching the MENU key. One of the interesting features of the configuration file is a reminder service. The editor can be instructed to watch for changes in certain files and warn you if any occur (a box will pop up on the editing screen). Thus the editor can check every few minutes for incoming mail, or see whether a background task has finally completed.

*INed* is much more suited to a workstation environment than *vi* because of these configurable features, the multitude of special keys, the speed of response, and the documentation. Teaching this editor to new users will be a much kinder task than confronting them with the obscurities of *vi*.

## Communications

UNIX systems have been known for their communications prowess, and this one is no exception. Even as you read this, UNIX systems around the country are exchanging mail, transferring files, and requesting the execution of programs over telephone lines—all without human intervention. PC/IX has all of these communications capabilities built in.

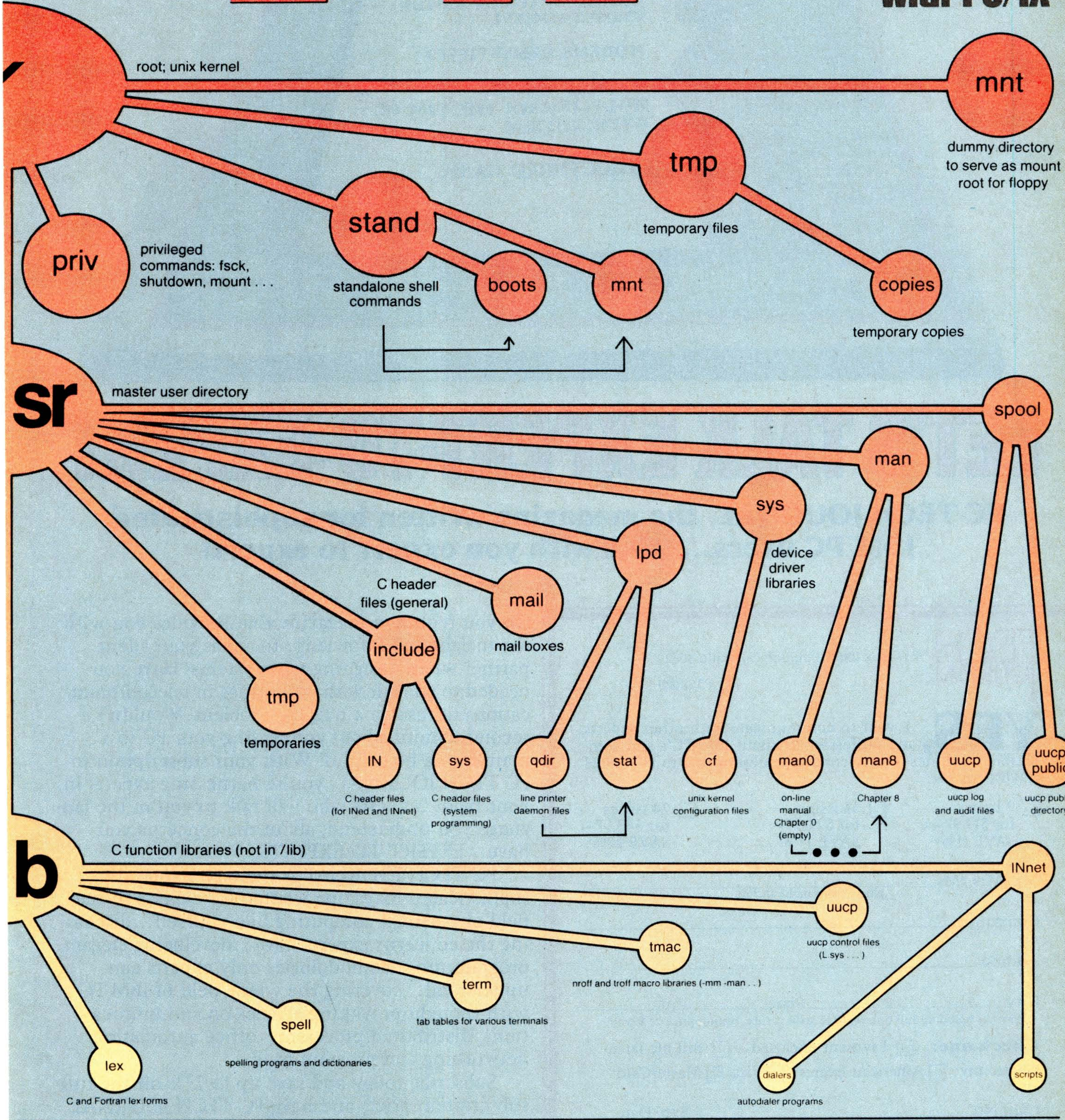
# PC/IX hierarchial directory structure



P C /
bin
major commands: cc, sh, mail . . .
dev
special files (devices)
etc
control files: passwd, ports, filesystems . . .
locks
locks for /etc files
lib
major C function libraries
lost+ found
u
tom
HOME directories for users
dick
harriet
adm
system accounting files and directories
acct
fiscal
nite
sum
bin
major commands not in /bin: awk, nroff, uucp . . .
games
games
lib
quiz
news
news
pub
n/troff files, ascii and ebcdic code maps
li
INed
forms
INed forms
help
INed help
helpers
INed helper programs
acct
system accounting programs
adb
adb scripts
cref
cref stuff
font
troff font libraries
help
help scripts

# IX

## Layout of the file delivered with PC/IX



root; unix kernel
mnt
dummy directory to serve as mount root for floppy
tmp
temporary files
copies
temporary copies
stand
standalone shell commands
boots
mnt
priv
privileged commands: fsck, shutdown, mount . . .
sr
master user directory
spool
uucp
uucp log and audit files
uucp public
uucp public directory
man
man0
on-line manual Chapter 0 (empty)
man8
Chapter 8
sys
device driver libraries
cf
unix kernel configuration files
lpd
qdir
line printer daemon files
stat
mail
mail boxes
include
C header files (general)
IN
C header files (for INed and INnet)
sys
C header files (system programming)
tmp
temporaries
b
C function libraries (not in /lib)
INnet
dialers
autodialer programs
scripts
uucp
uucp control files (L.sys . . . )
tmac
nroff and troff macro libraries (-mm -man . . )
term
tab tables for various terminals
spell
spelling programs and dictionaries
lex
C and Fortran lex forms

# PC/IX

*Continued from page 69*

The simplest communications are established with *connect*, a dumb-terminal program. There is a handy configuration file which allows *connect* to use the auto-dialing mechanism of some modems to call a particular system by keyword. Thus, if you execute

```
    connect oval_office
```

and you have an entry for *oval_office* in the configuration file with baud rate, phone number, etc., then there's some chance of connecting to the President's computer (don't tell him I sent you).

The powerful communications front-end program *uucp* allows the copying of files between machines. As with most other complex programs on UNIX, the workings of *uucp* are controlled by text files, governing abbreviations for other machines; how to reach them (phone, hardwire); when to reach them (middle of the night when the phone rates are low); how to log in, etc. Once the information about a remote site has been deposited into the control file (this need be done only once, and can be altered on demand by the system administrator), it can be retrieved by abbreviation.

To give an idea of how this works, let's suppose I want to transfer a file 'top-secret' to another directory, /usr/george, on my own machine; the command would be:

```
    cp top-secret /usr/george
```

If George has an account on the Pentagon computer, I can transfer the file to his directory on that computer by:

```
    uucp top-secret
              Pentagon!/usr/george
```

**For multitasking on the IBM PC, UNIX is the system of choice.**

Thus the invocation of the inter-machine copy program looks virtually identical to a local copy request. The copy program will look up the abbreviation for *Pentagon* in the control file, and find that that machine should only be called in the dead of night, with a particular login ID and password.

It will then stash the job away until the appropriate time to call, and when it does call, it will keep an account of whether it succeeds or why it fails—all without necessitating human intervention. And if it fails, it will try again. Of course, one can also request files to be transferred from remote machines to one's own machine!

Built on top of *uucp* is inter-machine mail. It looks, smells and tastes just like sending mail to other users on the same machine, but instead of

```
    mail george <top-secret
```

we send inter-machine mail with:

```
    mail Pentagon!george <top-secret
```

The mail program will notice that it should be worrying about inter-machine communications, and will pass the job on to *uucp*, which will do its dance of looking up *Pentagon*, etc. Meanwhile, the sender can go on to other tasks. It is even possible to request that the mail be passed through intermediate machines on the way to a final target (this is useful if there is a cluster of machines with hardwired communications, but only one of them has access to outside dialin or dialout lines). Indeed, one can get very fancy.

Last, but not least, there is inter-

## A Single-User UNIX?

No. This is definitely a multiuser implementation of UNIX System III. Although IBM markets the product to be installed as a single-user (but, of course, multitasking) configuration, it is relatively simple to bring it up in true multiuser mode. Those who know UNIX will recognize that any system which has the capability to run both incoming and outgoing **uucp** must have ordinary login procedures.

There are enough hints in the documentation—and, there is even code in configuration files that simply needs to be uncommented—to activate one or even two login ports. (In particular; the only file requiring modification is */etc/ports*, wherein the devices, */dev/tty0* and */dev/tty1* can be enabled for login. The *Operations Handbook* is quite clear on how to do this.) There is also help for configuring modems by several manufacturers for both auto-dial and auto-answer.

There are several reasons why IBM might be shipping the product as if it were only single-user. First, there is the official IBM reason (from Nadine Fletcher of the Information Systems Division): they would never put out a multiuser system without memory protection. Anyone writing in assembly language has free access to all of the physical memory of the 8088 and, therefore, to the operating system code. This makes security hard to enforce, since it is very easy to crash the system using assembler. IBM's official line (according to Ms. Fletcher) is that they will support PC/IX only in single-user mode, but purchasers of the system can do whatever they want.

Another reason for single-user: they have to pay AT&T significantly more in royalties for multiuser sublicenses. Finally, they may feel that the PC's 8088 running at 5 MHz does not have enough horsepower to support more than one user adequately, and do not want to encourage the buyer to expect otherwise.

IBM chose a particular type of screen editor to emphasize the single-userhood point. Since they did not expect other users to be logging in, they didn't need a terminal-independent screen editor like *vi*; in fact, Interactive Systems built an editor that interacts directly with the IBM video RAM (*INed*). Thus, not only is *vi* missing, but if an external user attempts to execute *INed*, the result is screen output to the system console! Consequently, it is strongly recommended that external users be prevented from using *INed*. For them, there is always the trusty—but unfriendly—line editor *ed*.

Then, the legal question with respect to AT&T arises: can one legitimately use PC/IX configured as a multiuser UNIX? There is nothing in the manual on such restrictions (in fact, the *Operations Handbook* is clear: "Assuming a port has been made to work as a device, there should be no problem enabling it for logins.") The license agreement simply prevents one from using the software on more than one machine at a time (!). This not only seems to permit multiuser usage, but even multimachine usage (provided only one machine is going at a time). The implication clearly is: if you want to, you can. I'm sure that if the product becomes very popular, AT&T will have something to say about this.—*Peter Brooks*

*Continued from page 73*

machine command execution (*uux*). One can request that a command, or sequence of commands in a shell script, be executed on another machine and that the output be sent back. This is similar to batch submission of jobs to mainframes, except that one can send this request to any other UNIX machine!

Now, this may all seem a bit fast and loose in terms of security. The fact is, that each system is as secure as its passwords. If you cannot log in on a remote system because you don't know any of the passwords, then neither can your local communications programs do that job. In other words, *uucp* cannot transfer any files to or from a remote system which needs passwords that it does not have. Furthermore, UNIX systems can limit the type of jobs they will execute for other systems (via *uux*). All of these limitations and capabilities are contained in a few control files edited by the system administrator—easily changeable, very powerful, yet quite secure.

### Languages

Naturally, the system comes with a C compiler. I have had no problems porting code to PC/IX, and large modules have been created automatically with *make*. It is difficult to benchmark the performance of the compiler, since there are few UNIX systems as small as this one. See Figure 1, however, for the results on the Sieve of Eratosthenes, an apparent de facto comparison, since it has been run on over a hundred computer systems (see Gilbreath: "Eratosthenes Revisited," *Byte*, Jan. 1983).

The IBM PC's processor architecture can make life difficult for your average C compiler. The 8088's memory segmentation, which deals with objects of up to 64K easily, has to work quite hard to deal with larger ones. Currently, the C compiler cannot handle very large programs—those whose code space or data space is larger than 64K. The linker/loader can separate data and code spaces into 64K each, yielding program sizes up to 128K, and can produce shareable code. Anyone interested in writing truly monstrous programs will have trouble, since the loader does not have the capability to generate overlays. One possibility is to split a large program up into separate, smaller programs called sequentially, or into simultaneous processes connected by pipes. There are other C compilers for the PC which can handle larger programs (e.g., Lattice C under MS-DOS); however, I don't know whether they run under PC/IX yet.

All of the standard System III C programmer's tools are here: *lint*, the C program checker; *cref*, which makes cross-reference listings; *adb*, the symbolic debugger; *cb*, the C program beautifier. The assembler, like most other UNIX assemblers, has sufficiently inadequate documentation to discourage usage. The only way to learn it is to analyze the assembler code output of the C compiler.

***INed* is easier to learn from scratch than *vi*, because many simple commands such as cursor movement are dedicated to special keys.**

There is an interactive compiler/interpreter called *bs* which combines features of both Basic and SNOBOL 4. It includes structured flow control (if. .then. .else, for/while .next), mathematical functions, undimensioned string variables, regular expression pattern matching, and file access. Like Basic, it can be used as an interpreter, where a line of code is executed as soon as it is typed in; or a program written in *bs* can be compiled for faster execution. It was designed for quick jobs where shell scripts don't have the necessary flexibility, and the full C language is overkill.

The developers of UNIX had a preoccupation with the preparation and manipulation of text files. Hence, many of the algorithmic languages shipped with PC/IX are dedicated to the transformation of textual files. The foremost of these is *nroff*, the text formatter. This is actually a full-blown formatting language, complete with user-defined macros, although a prepared macro library, *mm*, is provided for normal use. The macro library *ms* is not included, for unknown reasons.

If high-quality output is desired, the phototypesetting formatter, *troff*, takes the same input as *nroff*, but can drive a variety of typesetters. For instance, a mere $10,000 will buy an IMAGEN laser printer/processor that will take output from *troff* and print camera-ready copy in a variety of typefaces and sizes.

Along the lines of text manipulation languages, there is an interpreter/compiler called *sno* which is similar to SNOBOL 3, with some limitations. The pattern scanning and processing language, *awk*, is extremely useful for text table processing, as well as for a myriad of other small jobs. A more powerful and difficult tool than *awk* is *lex*, a lexical parser useful in combination with *yacc* (yet another compiler compiler), which will actually generate a compiler for a user-defined language. In fact, *awk* was created using *lex* and *yacc*.

There are more tools and small languages on the system than can be described here. However, no major programming languages except *C* are shipped with the system. In particular, **f77**, the Fortran-77 compiler, is missing (possibly because the compiler is too large for the PC?).

I expect that a variety of compilers will migrate to PC/IX shortly. IBM expects to offer **INFort**, a Fortran compiler for $250, in July.

### Special features

There are a few tools built specifically for PC/IX: utilities for reading, writing, and deleting PC-DOS files on either the hard disk or floppies, and viewing PC-DOS directories. A special misfeature is the necessity to bring the system down—into a state known as the 'standalone shell'—simply to format diskettes.

There is a safer form of the file deletion command, *del*, which requests confirmation before erasing anything (this can be a lifesaver). Error messages from the system are printed in highlighted characters on the 25th line of the console, which does not scroll off. This is a good practice, since these messages can occur during periods of heavy screen traffic.

### Support

You are likely to find very little support from your neighborhood software dealer (including the local IBM product centers). The product is too large and complex, and the dealers seem to know virtually nothing about UNIX. Your first job is, therefore, to convince the dealer that you want to talk directly to IBM, and that he should be able to find ("I'm sure it's around here, somewhere. . .") and give you the 800 num-

# PC/IX

*Continued from page 74*

ber directly to IBM's PC/IX product support group in Texas. *(Purchasers of PC/IX are entitled to the 800 number; however, IBM has made the dealer responsible for providing it. If you need support, make sure your dealer provides the number—Editor.)*

The folks in Texas are still easy to reach (as of this writing), and are quite friendly. They have a sort of triage system, where the person who initially receives your call will try to rate the severity of the problem. There is a scale from 1 to 4, with 1 being 'system-threatening' (it's down and you can't get it started), to 4 which is 'informational' (why did they leave out the chess program?). Problems of type 1 and 2 should get immediate response, with decreasing attention paid to 3 and 4. They also have a database of problems, and keep track of solutions.

**Since PC/IX may be too complex for dealers to support, there is an 800 number.**

And they do call back! Regularly! On the other hand, they were not able to reproduce any of my problems due to hardware difficulties of their own, and I have yet to get answers to some of my informational questions. But, at least there are live, friendly human beings to talk to.

Since they had just recently released the product, as of this writing, it's not surprising that few problems had as yet shown up, and that they therefore didn't recognize some of the difficulties I had encountered—a bad interaction between PC/IX and the Hercules Graphics board, for instance—and, thus, incorrectly suggested switch settings for the Hayes Smartmodem 1200 as a solution.

Given the likelihood that the product may be complex for their dealers to support, the idea of an 800 number deserves applause. I hope it doesn't get jammed as PC/IX becomes popular, and that they develop more experience with common problems.

## Summary

For those seeking a multitasking (and possibly multiuser) operating system on the IBM PC, UNIX is the system of choice. With tens of thousands of installations and hordes of programmers who know and love the system, UNIX is expected to dominate the operating system market in the 1980's.

PC/IX is a strong implementation of UNIX System III, with a few deficiencies (no *vi* or C-shell). The editor is simple and convenient to use, and almost all of the tools familiar to those who have worked on UNIX before are shipped with this version (700-800 files on the distribution disks). The system is slow when performing simultaneous jobs, but is adequately quick on interactive tasks. And the distributor, IBM, seems to have the financial wherewithal to remain in the software business in order to be able to provide support—at least for the near future.

*PC/IX is available for $900 at the IBM Product Center in your area.*

*Peter Brooks, 525A 6th Ave., Brooklyn, NY 11215*

**System requirements:**
IBM PC or XT with:
1 dual-sided floppy disk
10 MB hard disk
256K RAM
monochrome and/or color monitor

**Optional hardware supported:**
up to 640K RAM total (512K recommended for multitasking)
8087 math processor
3 additional single- or dual-sided floppy disks
1 additional 10 MB hard disk
2 serial ports
2 parallel ports

**Software modules** *(all optional except core):*

| | |
|---|---|
| 3.4 MB | core system (required) |
| 1.3 MB | programming subset *(adb, as, cc, lint, lex, yacc . . .)* |
| .3 MB | communications *(connect, uucp . . .)* |
| .3 MB | source code control system *(sccs)* |
| .4 MB | text processing *(nroff, tbl, mm..)* |
| .2 MB | special purpose *(troff, eqn . . .)* |
| .2 MB | system accounting |
| .3 MB | games |
| 6.4 MB | |