



The HP Microcomputer Journal

THE INDEPENDENT HP MICROCOMPUTER JOURNAL FOR SERIES 80 COMPUTERS

ISSUE #9 - OCTOBER - DECEMBER 1983

IN THIS ISSUE

THE HP-86/87 ASSEMBLER ROM AND MANUAL review	2
by David Efron	
HP-86/87 BREAKPOINT SUBROUTINE program	8
by David Efron	
SORTING IT ALL OUT article	12
by Don Person	
HIDDEN POWER CORRECTION article	16
by Don Person	
THE HP-150 TOUCH SCREEN COMPUTER review	18
by Dale Flanagan	
RAMBLING comment	21
by Dale Flanagan	
A NEW BINARY FOR PROGRAMMERS review	22
by Don Person	

(C) Copyright 1983 - Joseki Computer Corporation. All rights reserved.

NEWS80S is pronounced "News Eighties." It is published every other month by Joseki Computer Corporation, Redondo Beach, CA. 90278.

Editor: Dale N. Flanagan

Contributing Editors: David Efron, Don Person

Distribution: P. King

There is no relationship between NEWS80S and the Hewlett-Packard Company, except for a common interest in the growth of the Series 80 community.

ADVERTISING RATES are available by calling or writing for our current rate card.

The HP-86/87 ASSEMBLER ROM and MANUAL - Review by David Efron

The brochure is enticing, and the first exposure to the Assembler ROM induces the expectations of a child on Christmas morning. The ROM and manual are nicely packaged, as with all Series 80 products. The manual is professionally printed, on quality paper, with eight tabbed sections. The first few paragraphs are motivating, and the first skimming of the contents is encouraging. There are diagrams, tables, sample programs, and entry-points for using a truck-load of system routines.

On second reading, a certain realization develops. It starts with the caveats and warnings in the Introduction. "The manual assumes you have some knowledge of programming in assembly language." Naturally. . . but how much? To many, the system's architecture is different, and its instruction set is alien. The system's internal routines are well-documented, but how are they used? Indeed, can you understand the documentation? The executive, initialization, the parser, the interpreter, interrupts, and hooks; all the pieces fit together, but where do you start? Do you have to go through all that just to write a program which adds up a long column of numbers? The questions mount up, but not as fast as the anxiety.

A time-lapse occurs . . . for fast learners and quick-studies probably not so long, but ego and professional pride prevent me from revealing a yardstick. Nevertheless, it finally happens. The onion peels away, layer by layer. The eyes open up, wider and wider. The promises seem tangible, the jig-saw is not so much a puzzle.

This is not to say that somehow the difficult becomes easy. It is rather that most of what is needed is contained in the manual, and that it becomes clear after much digging.

THE ROM AND ITS COMMANDS

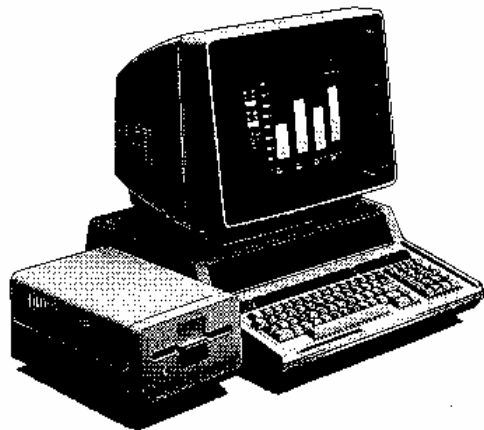
As a plug-in ROM, the assembler and its commands are always available. The diskette provided with the system contains sample programs as a convenience, but otherwise it is not needed.

One command invokes the system, and another gets back into BASIC. The transition into Assembler mode is easy, and the new environment brings only a few changes.

Source files are *ASTORE*'d and *ALOAD*'ed, but that's about it. *AUTO* line numbering and *RENumbering* are the same, as are the cursor movement keys and line editing keys.

Two new commands are offered to help search through the source code for occurrences of a string-value. *FLABEL* finds a statement label and *FREFS* will find any string, whether it is in a label, opcode, address field or comment. The *FREFS* command also works in BASIC, where so far it has received a greater workout.

Entering lines of source code involves a few simple format conventions. One space after a line number (provided by *AUTO*) is required. A label starts in the next column, with up to eight characters. If the line has no label, leave an extra space, or else the opcode is taken as a label. After the instruction, allow one space then an exclamation point to signify a comment. The Assembler automatically formats the line with the label left-justified in its field, the instruction left-justified 10 print-columns from the statement number, and any comments 34 print-columns from the line number.



announcing FILE/IDEA the new data base management program for the HP series 80 computers

FROM THRESHOLD SOFTWARE INC.

introductory priced at only \$69.95

FILE/IDEA
gives the HP series 80 a way with
your facts and figures

FILE/IDEA is the new data base management program from Threshold Software. Its rapid information storage, retrieval and manipulation allows for alpha, numeric and dollar data. It includes sort-merge utilities which interface to other programs such as: WRITE/IDEA, PaperRiter, Calc files and plotting programs, the HP Standard Regression Analysis Pac, or even many custom developed programs.

FILE/IDEA is so easy to master, and has interactive screen prompts that are so friendly and direct, that you will quickly find yourself conscious only of the data, not the computer. In conjunction with built-in "utility" programs, FILE/IDEA expands your HP series 80 into new application domains by maintaining active record files in business, engineering, or research applications.

FILE/IDEA features:

- Alpha, numeric (floating point), and dollar (2 decimal place) data.
- Interactive file creation to allow file and field optimization
- Indexed file and quick FIND function to locate records.
- Display of entire record on screen and full screen editor.
- Quick record search or select, plus select by alpha substring
- Searching and sorting treat numeric & dollar data as numbers.
- Find, delete, sort, merge, or print selected records
- Extensive error-trapping and English message interactions.
- Reconstruct index in case of accidental file corruption.
- Extremely flexible printout formats.
- Powerful utility programs to allow
 - initializing the disc for optimum file size
 - up or copying files, even with only one disc drive.
 - generation of form letters in conjunction with the WRITE/IDEA program.
 - selective redesign and merging of files.
 - ascending or descending multi-level sort
 - true algebraic value numeric sort.

FILE/IDEA specifications:

64k minimum system with one disc drive. More drives and/or Winchester are ok. Maximum file size: 2000 records, 20 fields/record, 64 characters/field, 10 character field names, 261,000 bytes/file (with one file: 3.5 or 5.25 disc).

introductory priced @ \$69.95

WRITE/IDEA puts in a friendly word for your HP series 80

WRITE/IDEA is an extremely user friendly, highly flexible, responsive word processing program that quickly and substantially enhances the productivity of your HP series 80 personal computer.

WRITE/IDEA is designed to swiftly and easily produce personal and business correspondence, client lists, reports, "personalized" form letters and most general purpose written communications. These capabilities are logically assigned to the HP keyboard which allows you rapid mastery of the program and its special features. The high degree of user friendliness engineered into WRITE/IDEA assures immediate and total operator fluency.

WRITE/IDEA features:

- Insert or delete a character, word, sentence, paragraph or block of text.
- Search & replace globally or selectively.
- Line formatting left, right, centered, or justified
- Text scrolling and direct jump, copy & paste, decimal tabbing, merging.

WRITE/IDEA requirements

32k bytes of memory and, in the case of the HP 85 the printer-plotter ROM.

specialty priced @ \$69.95

also available from
Threshold Software:

PapeRiter

PapeRiter is a word processing program for the HP 86/87 specifically designed for technical paper, scientific report, and documentation applications. PapeRiter has the ability to use five print type styles, number pages with Roman or Arabic characters, place footnotes, and incorporate superscripts, graphics or special characters (such as arithmetic symbols) with text. PapeRiter regards the screen and print copy as equivalent so that "what you see is what you get." All this in addition to the expected insert, delete, search, replace and other functions of a fully text oriented word processor. PapeRiter is interactive with FILE/IDEA. It requires 64k RAM, one disc drive for hard copy and is compatible with most marketed printers.

introductory priced @ \$99.95

COSAC

A powerful job costing and predicting program for engineering, contracting, manufacturing and service organizations that follow costs on a job-by-job basis. COSAC pinpoints productivity characteristics and how each job is doing. Essential for quick, accurate estimating and profit forecasting.

specialty priced @ \$69.95

COSAC PLUS

The same fully featured job costing and predicting programs as COSAC with the addition of an accounts receivable function. Complete reporting formatted in hard output reports that will vastly assist project control and management. Monitors work-in-progress, handles accounts receivable in a paged journal, generates invoices automatically.

specialty priced @ \$99.95

SERCAL

A program that calculates the size of electrical services to commercial buildings. SERCAL produces forms that are designed for submittal as well as other forms that aid designers who are considering alternative ways of providing power service. Calculations are traceable to the National Electrical Code and sizing is by trade practice.

specialty priced @ \$69.95

ORDER FORM

These prices are effective until November 30, 1983 and write-in support for all programs is limited to product problems.

If media format is not specified 5.25" disc will be sent.

Please enter my order for the following program(s)

____ FILE/IDEA (HP 86/87 only)	disc size _____	\$ 69.95ea	
____ WRITE/IDEA	disc size _____	\$ 69.95ea	____ tape cty. \$94.95
____ COMBINATION OFFER			
____ WRITE/IDEA & FILE/IDEA	disc size _____	\$125.00set	
____ PapeRiter (HP 86/87 only)	disc size _____	\$ 99.95ea	
____ COSAC	disc size _____	\$ 69.95ea	____ tape cty. \$94.95
____ COSAC PLUS (HP 86/87 only)	disc size _____	\$ 99.95ea	
____ SERCAL	disc size _____	\$ 69.95ea	____ tape cty. \$94.95
total enclosed \$ _____		\$ _____	
(California residents add 6% sales tax)			

ship to:

name _____

company _____

street _____

city _____

state _____

zip _____

telephone (_____) _____

IMPORTANT! Prices extended until January 31, 1984!

THRESHOLD SOFTWARE 1832 Tribute Road Suite E Sacramento CA 95815

(916) 920-8189

A SCRATCHBIN command is included in the ROM, and several functions are also included for use in BASIC and Assembler modes. OCT converts a decimal number to its octal equivalent. DEC converts octal to decimal. For some of us, the system could be improved by including HEX functions, to-and-from both octal and decimal. However, it should be noted that while the HEX functions could help a lot in the beginning, in time the octal notation becomes just as natural.

There are also two memory-dumper commands, MEM and MEMD. In the first, supply an octal address and the system will dump octal values and their ASCII equivalents. In the second command, provide an octal address where a three-byte address (low to high) is found. The system will dump memory from that address. Both these commands also allow the programmer to change the contents of memory.

(New users please note: The manual does not accurately show the syntax of the OCT and DEC functions. Whereas MEM 101010 and MEM (101010) both will work, the OCT and DEC functions must have parentheses enclosing the value.)

USING THE ASSEMBLER - (AN ASIDE ON THE TECHNICAL SIDE)

The instruction set is overwhelming at first. Compare it to the sparse 6502 chip's instruction set, for instance! Then look carefully and realize that there are really nine fewer instructions than shown, because of a typesetting error which inserted a section of the instruction set redundantly. Then realize that it's not all that much to learn, because there are many addressing modes which explode the skeleton instruction set.

The Assembler, because of the system's architecture, can operate on as many as eight bytes at a time. This makes for compact programs. Again, a comparison: Add 1 to a two-byte address, or to a counter. It's as simple as that. Never mind about incrementing the low-byte, watching for zero, then incrementing the high-byte if necessary. Handle three-byte addresses just as simply.

Two nitty-gritty examples of what might be trouble-some are the stack operations and addressing modes. For one thing, it is often difficult to tell which is the top and which is the bottom of a stack, and what is its origin. As for addressing modes, careless typing can bring subtle errors. LDMD R20,=LOC means bring me the two-byte value at the address called LOC. If by mistake the instruction were typed as LDM R20,=LOC, the Assembler would accept it as a good instruction, but the program will hand over into registers 20 and 21 the address which LOC refers to, not the two bytes resident at that address. (It is probably safe to say that all assembly languages have similarities of this kind, but this one is mentioned as an example of an oft-repeated mistake that seems to defy proofreading of source code.)

Jumps within a program are made to label values, within +127 or -128 bytes. With all but one status bit, jump instructions are available on either the zero or one condition. Subroutine jumps can be anywhere within the first 64K, where all binary code must reside. Absolute jumps beyond 127 bytes are discouraged in the text of the manual, yet they are used in the sample programs. The GTO pseudo-instruction allows these broad-based jumps, but this programmer refuses to use it, after reading a technical note saying that there is a bug in the Assembler and sometimes it GTO's the wrong place. Instead, load the program counter yourself.

Very long programs can be segmented, if too big for RAM. Start assembling the first

segment, and when a LNK “filename” instruction is encountered, the Assembler will load that file and continue assembling.

Object code is normally not printed, but the LST instruction causes it to be printed to the PRINTER IS device until an UNL instruction is encountered.

The Assembler also has a code-optimizing feature. Where successive instructions use the same data register and address register, the Assembler removes the redundant “set data register pointer” and/or “set address register pointer” instructions, which would otherwise be put in.

Immediate feed-back is provided on the syntax of instructions. Too-long JNP’s generate error messages at assembly time. If successful, the assembled binary code is automatically stored on the disk, all binary programs in memory are scratched, and the new binary program is loaded. However, the programmer can override the scratchbin and loadbin parts of the ASSEMBLE command.

To buy the System Monitor ROM. . . or not to buy. . . would that it were in the ROM drawer at this moment! It lets the programmer set two breakpoints: memory addresses which, if referenced in any manner, cause a pause in execution and a dump of registers and status flags. The ROM also has STEP and TRACE capabilities. STEP executes one instruction, dumps registers and status flags, and pauses. TRACE will dump the status flags after every instruction without pausing.

This reviewer did not buy the Monitor ROM, but has regretted the decision. The STEP capability alone is worth its price. However, born of necessity, a simplified breakpoint subroutine was developed to capture the status flags, registers, DRP, ARP, and Extend register. It is printed following this review, both as an aid for others who failed to buy the Monitor ROM, and as an example of what the Assembler code looks like. It’s output is not as nifty as the Monitor’s, but it might help in a pinch. (It could be modified to dump its save-area to the screen or printer, thereby allowing something like the STEP or TRACE features.)

WRITING PROGRAMS WITH THE ASSEMBLER

A previous article in NEWS80S has described the “shell” of a binary program. This is a set of required program modules, linked to each other and to BASIC through rigidly defined pointers built into the system. Once the concept is understood, the programmer can become very productive, since the system performs much of the overhead of a typical binary program which has to test for the validity of parameters passed to it.

However, the writing of the parsing module, and the use of the system’s parse routines, could pose the most serious learning exercise. Before the first program line can be written, the programmer first has to learn how the Parser parses a BASIC statement; how a tokenized statement is decompiled; how to use the parse-routines; the relationship between RPN logic and the operating stack, and how variables are passed on the operating stack.

Never mind all that for the moment. The system will do the parsing automatically, if the binary program is defined to be a “function” as opposed to a “BASIC” statement. Functions can have one or more string and/or numeric arguments. Tell the system how many of each, and in what order they will appear, and it will worry about parsing them, finding them in memory, and passing them to your program at the right time. For example, X = HEX (OCTAL) could be a numeric function with one numeric parameter. A little fancier example is something like

X\$ = CONV (N\$,B,NEWB), which could be a string function to convert a string value of a number, expressed in base B, to a number in a new base NEWB, i.e. decimal to octal, binary to octal, octal to hex, or anything else.

To write a BASIC statement, which might have optional parameters, the parsing must be done by the programmer, and that requires study of the parsing routines and examples. Expect to wear out the corners of many pages of the reference manual, flipping between descriptions of the system's routines, the description of the operating system, and the description of BASIC variables and values. This is when first-base seems to be the combined distance of first, second, and third.

When you get there, though, the system can be called a work of art. At that moment the potential of the system can be seen as it is described in the opening paragraphs of the manual. Then in the next moment a new realization sets in, and the reason to become critical of the Assembler package becomes clear.

THE MANUAL AND REFERENCE MATERIAL

At first it's great. The architecture of the hardware, the operating system, the interrupt capabilities, the structure of BASIC programs, the CRT and keyboard interfaces --- it's all there.

Then it's missing something: Mass storage routines, reserved locations you'd like to use, source listing of the system's routines. The sample programs utilize "tricks" which can be devised only by having knowledge of the source code of the system's routines. Branches are made from the sample programs into the middle of system routines, but the how's and why's are not detailed. After going to the trouble to understand the logic of the sample programs, one is left with the feeling that the potential for using the Assembler is great, but only for those inside "The Company".

And then there are errors: Many are typo's, but many are errors. In either case, the potential for great programs grows more distant. For example, the reference material might say that before using a certain routine, the programmer must first call a subroutine to switch ROM banks. That may, or may not, be true. For the subroutine which outputs a string to a printer, the documentation says that the Mass Storage ROM first has to be selected, i.e. the routine is on that ROM. Remembering the caution at the outset of the manual, that "it is also possible to defeat the computer's internal safeguards and even seriously damage the computer", one must be thankful that an expensive peripheral did not detonate upon executing the code which branched into some uncharted region of the Mass Storage ROM's instructions.

After some time, one can begin to detect the errors in the manual beforehand, but only after becoming familiar with what is on the different ROM's. Nevertheless, the quantity of errors instills a certain lack of trust and a certain level of fear. Aside from these technical flaws, one can further criticize the manual for flaws in its content.

The manual does not centralize information on a given topic. References are made here and there, and a cross-reference or index of key words is not provided. A case in point: Page 3-22 offers a segment of code to steal a "hook". Use that code as shown, and all is well; but get creative and try using a different set of registers, and there is going to be trouble. The choice of R34 in the example is not whimsical -- but it is not explained until page 3-27, and without cautionary fanfare, that use of

any registers other than 34-37 might conflict with the system's use of registers.

There are source-code listings for five programs, all of which are contained in assembled form within the UTIL/1 binary program collection on the System Demo Disc. They are an appropriate set of programs, arranged in order of increasing level of difficulty. All are heavily commented for educational purposes, and studying them provides the glue that binds all the pieces of information presented by the manual.

Still, there is room for enhancement. Considering some of the "tricks" used by the authors of these programs, the explanations are a bit lean. More source code would certainly help; while the documented examples are certainly a treasure, the learner is surely left hungry for more --- especially when the example programs make branches into undocumented routines where only the source listings would shed an adequate amount of light on the subject.

IN SUMMARY

Patience and potential, one leads to the other. The Assembler ROM, with the means for creating binary programs and patching them into BASIC programs, are all well-designed and within reach of a programmer versed in assembly programming techniques. The potential for enhancing BASIC programs, with speed improvements and code-saving intrinsic functions, is realizable. The potential grows as the programmer's familiarity with the system increases.

But caution is required. For other than the assembly-expert, there is a lot to learn, and a lot to get through. Errors in the reference material, and the manual of instruction itself, make the job more difficult.

Nevertheless, the information regarding the make-up of a BASIC program itself may, for many, be worth a large part of the cost of the package. For the dedicated, and the fast-learners, the Assembler ROM and manual will reward their efforts, without doubt.

----- WANT ADS

FOR SALE: 11P85A, ROMS, Interfaces, Software, 82901M Dual Disk, 82905A Printer, 7225B Plotter at 50% list price. Upgrading to Series 200. Shannon Davis, P.O. Box 813, Corvallis, OR 97339 (503) 758-1432.

FOR SALE: Have converted- will sell HP-83A, HP-85A, 16K memory, HPIB, ROM Drawer, ROMS, Application Pacs, 82949A Parallel Printer Interface, and 9111A Graphics Tablet. All in mint condition at half price, much of the above never used. (313) 528-1571 or P.O. Box 1034, Troy, MI 48099.

FOR SALE: HP-85A with 82901M Dual Disc Drive, 32K memory, Advanced Programming ROM, Assembler ROM, HPIB Interface, and other required hardware. John L. Brown, 66 Orange St., Brooklyn, NY 11201 (212) 834-9244.

Series 80 (for 85,86,87): 82909 RS-232 Interface \$300; IDS 560G 200 cps printer \$850; 9130A 5.25 Disc Drive (for 86A) \$580; (for 86,87): 85-13058 Statistics Pac \$170; Write/Idea word processor \$120; 85-13044 Data-Comm Pac \$150; dBase II \$360; Randy Webb, 622 East 11th St., Bloomington, IN 47401 (812) 339-7661.

FOR SALE: HP Mass Storage ROM \$55; HP I/O ROM \$110; HP 16K Memory Module \$110; All for HP-85. George J. Hofer, P.O. Box 132, Platte, S.D. 57369 (605) 337-3038. Or all for \$260.

HP-86/87 ASSEMBLER BREAKPOINT SUBROUTINE by David Efron

The following assembly language routine is meant to be used as a program development and debugging tool for HP-86/87 programmer's who don't own a systems monitor ROM. Since this is a subroutine, it doesn't have the "shell" that is necessary to create a full Series 80 Binary program. The comments in the code explain what it does and what the output should look like. After debugging of the binary program, references to this subroutine and the subroutine itself can be eliminated and the binary can be re-assembled into a "production" version.

```

9000 BRKPT   SAD                ! Save before we do anything
9005        PUMD R70,+R12       ! Save the 70's, 60's and 20's
9010        PUMD R60,+R12       ! because we have to use them
9015        LDM R70,R20
9020        PUMD R70,+R12
9025        BCD
9030        CLB R22             ! Shift E into a zero'd byte
9035        ELB R22             ! (four bits at once in BCD mode)
9040        BIN
9045        PUBD R22,+R6        ! Put E onto stack
9050        PUMD R70,+R6        ! Save the 70's again, so we can use them
9055        LDM R22,=BRKTABLE    ! We need R22-23; they're clobbered,
9060        STM R22,R75         ! but we save them above.
9065        CLB R77             ! Set up for PTR2 stores
9070        ADMD R75,=BINTAB     ! (The absolute address of our save area)
9075        STMD R75,=PTR2
9080        POMD R70,-R6        ! Now . . . Save the 70's in the save area
9085        STMI R70,=PTR2-
9090        STMI R60,=PTR2-     ! And the 60's
9095        STMI R50,=PTR2-
9100        STMI R40,=PTR2-     ! The 50's and 40's
9105        LDM R70,R30
9110        STMI R70,=PTR2-     ! For the 30's on down to the 0's, first
9115        POMD R70,-R12       ! move them to the 70's, EIGHT registers
9120        PUMD R70,+R12       ! at a time.
9125        STMI R70,=PTR2-     ! The 20's came off the stack, but went
9130        LDM R70,R10         ! back on for later restoration
9135        STMI R70,=PTR2-
9140        LDM R70,R0
9145        STMI R70,=PTR2-
9150        CLM R60             ! Zero out some space
9155        POBD R67,-R6        ! Get the E register back
9160        POMD R75,-R6        ! Get back the SAD (arp,drp,status bits)
9165        LDM R20,=77,77     ! Mask for the arp and drp
9170        LDM R22,R75         ! R75-76 have the ARP and DRP
9175        ANM R22,R20
9180        STB R22,R66         ! Save them
9185        STB R23,R65         ! in reverse order for the screen display
9190        STMI R60,=PTR2-
9195        CLM R60
9200        LDB R22,=1          ! To invert LZ,ZR,RZ (SAD inverted them)
9205        STB R75,R23         ! Move the ARP part of the SAD to get the

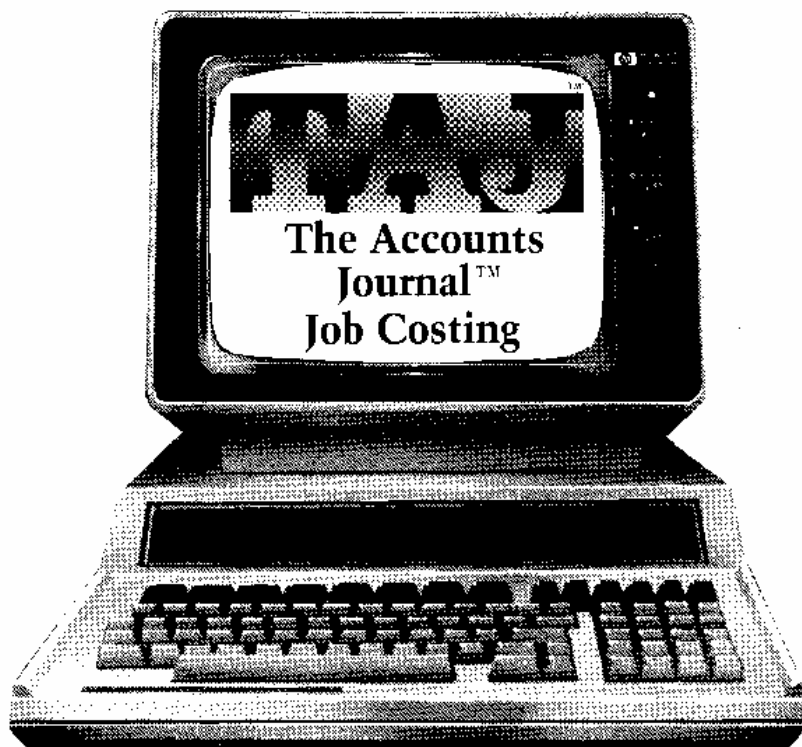
```

```

9210      ELB R23      ! OVF and CY bits
9215      ELB R60      ! Shift each one into the CY, then
9220      ELB R23      ! into one of the 60's, to end up with
9225      ELB R61      ! eight bytes of 0 or 1
9230      STB R77,R23  ! FOR NG,LZ,ZR,RZ,OD
9235      ELB R23
9240      ELB R62      ! . . . do the same thing
9245      ELB R23      !
9250      ELB R63      ! -----
9255      XRB R63,R22  !           S t a t u s   F l a g s
9260      ELB R23      !           OV  CY  NG  LZ  ZR  RZ  OD  DC
9265      ELB R23      !           ---  ---  ---  ---  ---  ---  ---  ---
9270      ELB R23      ! R00:000 000 000 000 000 000 000 000
9275      ELB R23      ! R10:000 000           000 000
9280      ELB R64      ! R20:000 000 This is what it 000 000
9285      XRB R64,R22  ! R30:000 000 will look like. 000 000
9290      ELB R23      ! R40: .   .   .   .   .   .   .   .
9295      ELB R65      ! R50: .   .   .   .   .   .   .   .
9300      XRB R65,R22  ! R60: .   .   .   .   .   .   .   .
9305      ELB R23      ! R70: .   .   .   .   .   .   .   .
9310      ELB R66      ! R80:000 000 000 000 000 000 000 000
9315      STB R76,R23  ! -----
9320      ELB R23      ! FOR DCM
9325      ELB R23      ! get passed the OVF
9330      ELB R67
9335      STMI R60,=PTR2- ! Save the 8 status bits
9340      LDMD R75,=PTR2- ! Save start-of-table in 100470 (octal)
9345      STMD R75,=100470 ! *** Use MEMD 100470,120 to look at
9350 !                   the dump of registers and flags
9355      POMD R70,-R12 ! Restore the registers we used
9360      STM R70,R20
9365      POMD R60,-R12
9370      POMD R70,-R12
9375      RTN
9380 ! ===== Use the subroutine as follows: JSB X22,BRKPT ! where R22-23 has
9385 ! ===== the BINTAB address. Then : RTN ! or continue with
9390 ! ===== the program
9395 ! ===== Finally, use the command MEMD 100470,120
9399 ! =====
9400 BRKDUMP BSZ 120
9405 BRKTABLE BSZ 0
9999 ! =====
10000 BINTAB DAD 104070
10010 PTR2 DAD 177714
10020 PTR2- DAD 177715
20000 FIN

```

Complete Job Costing And Integrated Accounting
Exclusively Designed For H.P. Series 80 Personal Computers By
Production Data Systems



TAJ™ Accounting

- TAJ™ understands and accepts English commands like "ADD EMPLOYEE" and "PRINT CUSTOMERS."
- TAJ™ is never out of balance. Transactions require both a debit and credit which are posted simultaneously.
- Processes and prints reports without an operator being present.
- Complete aging schedule includes a 60-90 day and 90 day + column.
- Automatic end of month, quarter, calendar, and fiscal year closing.
- Multiple pay periods and salary structures.

TAJ™ Job Costing

- Tracks direct and indirect (overhead) costs for labor, non-labor, and "other" categories.
- Detects cost overruns and overhead inefficiencies.
- No multiple data entries if being used with TAJ™ Accounting.
- TAJ™ can be used for bid preparation, cost control, and forecasting.
- Handles integration with TAJ™ Accounting automatically.
- Report detail is defined by the user to minimize paper usage but still provide the desired analysis.

For More Information



Production Data Systems
Sales Office
2386 Fair Oaks Boulevard
Sacramento, California 95825
(916) 484-0155



TAJ™ and The Accounts Journal™ are trademarks of Production Data Systems, Sacramento, CA

SORTING IT ALL OUT by Don Person

For the programmer who has to deal with the management of more than the most miniscule data-base, the issue of sorting is among the most important that you will have to deal with. In the case of the SERIES 80, this aspect of your endeavors is complicated by the speed of execution of your computer. For this reason, more than any other, it is of the utmost importance that you optimize the procedures that you use to accomplish this task, and in this presentation, we'll concentrate on the this.

This article is not meant to be a complete tutorial on sorting; for that there are a number of "how to" texts available of a generic nature that are applicable. The simplest method of sorting, and also the most conservative of RAM, is the bubble sort. Unfortunately, when coded in Basic, it is very SLOW. Array sorting methods consume operator time in a fairly linear progression as the number of elements sorted increases. Bubble sort times increase exponentially as the number of elements increases. This limitation makes the bubble sort, however implemented, most useful where relatively small numbers of items are to be sorted.

There is a binary tool that you can acquire which can be used as a very nice bubble sorter. The program I am referring to is the binary "SORTB2g". If you already have this one, you may have felt "burned" if you have not read between the lines of the rather minor documentation included. I would like to now present to you a method that utilizes the best aspects of the array sort and the bubble sort simultaneously. I'll show you how to sort individual items of data, both strings and numbers with this one.

In case the supplied documentation made little sense to you, here is a quick review of the applicable command in "SORTB2g":

```
SORT BUFFER$,8,1,6
```

This tells the system that the contents of a string called BUFFER\$ are specified for sorting. The 8 tells the binary that the contents are to be evaluated in groups of 8 at a time. If the length of the string is not a multiple of 8, then an error will be returned. The 1 tells the program to examine the specified block of eight characters starting with the first, and the 6 tells it to stop comparing at the sixth character. It is important to note that even though less than the entire substring can be evaluated for position in the buffer column, the entire declared length will be relocated (In this example, a block of 8 characters). A number of meaningful, custom error messages are included.

The usage of UPCSORT\$ should need no further explanation, as this keyword is pretty self-explanatory.

As an example of how this works, here is a test string with elements 8 characters long that you want to sort by looking at the first 6 characters.

```
BUFF$= "BEAKER76SERUMS15XRAYS 05"  
!----- !xx!----- !xx!----- !xx
```

Notice that the area between and including the exclamations is the area whose value will be sorted. In this example, the numerics could represent pointers to a file which could be used, after the content of the string as a whole is sorted.

From this you can see that if we can efficiently package data and pointers with a CONSTANT LENGTH into a string, we can use the SORT command found in the "SORTB2g" binary to rearrange the contents of the string in proper order. By using a loop, we could then read just the numeric pointers attached to direct us to the sorted data items.

One point I should make right here; if you later adopt this sorting procedure, when adding to an already sorted list, always insert new items at the head of the sort buffer string. The SORT command uses a bubble UP ONLY algorithm, so a single item added to the end of the list, destined to come to rest at the mid-point, will take as long to come to rest as it takes to sort a randomly ordered list of half the length present. The same item inserted at the bottom floats directly to its place in order with only 1 block of 'swaps'.

To properly add an item for insertion in our already sorted BUFFER\$, you would program:
BUFFER\$ = NEW_ITEMS\$ & BUFFER\$ @ SORT BUFFER\$,8,1,6

Here is an example program that will generate a group of random numbers and sort them using the SORTB2g:

```

10 OPTION BASE 1
20 INTEGER A(100)
30 DIM BF$[820] ! The sort buffer string
35 LOADBIN "SORTB2g"
40 ZERO$= "00000000" @ BF$= ""
50 DEF FNN$(X,Y) ! This function will turn a pointer and number into an
                   8 character string. The last 3 characters are
                   a pointer & the first 5 the number to be sorted
                   ie. 12345001 (integer is 12,345 & pointer is # 001
60 W=LEN(VAL$(X)) @ Z=LEN(VAL$(Y))
70 FNN$=ZERO$[1,, 5-W] & VAL$(X)&ZERO$[1 , 3-Z]&VAL$(Y)
80 FN END
90 ! NOW WE LOAD THE ARRAY WITH NUMBERS AND PUT POINTERS IN THE BUFFER
100 FOR L1 TO 100
110 A(L)INT(1E5*RND)
120 BF$=BF$&FNN$(A(L),L) ! PUTTING THE NUMBER AND POINTER PAIR IN BF$
130 NEXT L
140 ! NOW FOR THE REAL ACTION
150 SORT BF$,8,1,5
160 ! TO SEE OUR HANDIWORK WE DISASSEMBLE THE STRING, IN THIS CASE LETS
    PRINT THE SORTED LIST
170 FOR L=1 TO LEN(BF$)-7 STEP 8
180 PRINT A(VAL(BF$[L+5,L+7]))
190 NEXT L
200 END ! THE LIST IS PRINTED IN ASCENDING ORDER , TO PRINT IN DESCENDING
    ORDER, LOOP FROM THE TOP DOWN

```

This method requires 8 bytes per number sorted in the sort buffer string. In this regard, it's not exceptionally compact. What would be really handy would be a means of generating natural binary coded representations of both the pointers and the weighted data. In this manner, for instance, if we allowed 2 characters for pointers, we could point to 1 of 65k unique addresses. Six more bytes would be adequate to represent integer values in the billions.

Well, as it turns out, with a little bit of help from the FILE/80BIN program supplied with the HP FILE/80 program, this method can be expanded to be your sorting helper for data re-arrangement of numeric values. The FILE/80BIN "ICHR\$" and "ICH2\$" commands can be used to make a 5 character pointer that could represent blocks of zipcodes stored as strings (NOTE: See Don's article on "Hidden Power" in NEWS80S number 8 for a complete explanation of these and other commands - Editor).

The first 3 bytes of this 5 character string are the ICHR\$ representation of the zipcode, the last 2 bytes are the ICH2\$ conversion of the pointer. Using this type of conversion of the zipcodes and pointers, along with the SORT command and a few other tricks, 400 ZIPCODES can be sorted in 9.2 SEC. I have used this technique to sort blocks of up to 1200 zips so far.

Of course, this type of refinement is not needed for simple sorting jobs. If we wished to only some 5 digit zipcodes, the first method shown in the program listing above will work fine. It has the advantage of not needing the FILE/80BIN enhancements to work, and is still much faster than other sorts. Even without the extra enhancements, you are still way ahead with even the simplest Basic-only sort.

Before I demonstrate the application of the sort program to Alphabetic ordering, I thought I would review what many of us already know about sorting string values. Most versions of BASIC, HP included have the capability of evaluating strings of different lengths, two at a time, and determining the more heavily weighted string, even in situations where the length of each string is un-equal. This is frequently exploited in array-sort methods. As you are probably also aware, this is not particularly fast either.

In order to short-cut this procedure using the SORT command, we would have to produce substring units in our buffer column that were of constant length. This is unfortunate, because all strings would have to be of the same length, hence as long as the longest string to be evaluated. If the longest string was 30 characters long, we would have a total sub-string length of 30 (or more, assuming we used the ICH2\$ conversion for a pointer).

There is a way out, without giving up the extra speed we've just gained. The solution lies in the realm of a statistical solution. In most practical applications, many fewer than 30 characters need to be sorted to give a very low error rate in the sorted list. Theoretically, we are dealing with powers of 26 as we increase the number of characters sorted, but in the case of human names, common groupings of letters require more than the theoretical 6 character sort to achieve an error rate less than .003 %.

If we sort names using 8 characters, we can realistically expect better than .001% error in the sorted list. If we then append a pointer, using a "straight" string representation of a number or the ICHR\$ or ICHR2\$ commands, we can then sort as before. For relatively few entries, a single buffer is adequate.

I recommend organizing the sort task so as to never sort much more than 2000 characters in a single column. In the case of minimum memory systems, your best bet is to read just the sort fields in from disk, and then add pointers that will tell you the total record's location on the disk.

Although you can use UPMSORT on the alpha sort, I suggest performing the conversion on

the sort keys during the pointer generation routine. At the same time, you could remove any unnecessary spaces from within the name/word. The STRNGBg binary provides a simple way to remove spaces with SAR\$ command. The syntax for this command is SAR\$(string, match string, replacement string). For example: NAME\$ SAR\$(NAME\$, " ", "") will be remove spaces. Performing a TRIM\$ on all entered data would also be a good programming habit to acquire.

After sorting the buffer, you can then read the indicated disk files in pointer order. This allows for large sorts without optional memory, but has the disadvantage of being more disk intensive. Your requirements, and the configuration of your system should be your guide in deciding how to apportion system resources.

As a helpful guide, let me suggest a means for streamlining the sorting of human names. In business applications, it is one of the common sorts, and if it is an application for you, the following will be helpful.

When sorting names, use a buffer with 26 rows (one row for the starting letter of each name), and use the SORT command on each individual row. This is a little tricky, because the HP Series 80 is lacking any dynamic string allocation features, and we must dimension our string buffer arrays to a fixed length. To do this, you must know the average distribution of the first letter of last names.

I conducted an informal study of a number of metropolitan phone books from around the country to try to determine the average distribution of last names by first letter. I discovered that the letter "S" and "M" are the winners for starting last names, at 10% each. Thus, as a minimum guideline the sort buffer array row length should be no less than the length of an individual buffer entry times 10% of the total number of entries to be sorted. A nearly foolproof multiplier would be 15% (.15).

For example, if the length of the sort string is 8, and we will use a 2 character pointer, and we will sort 1000 names, and we would like an extra margin of safety, we would multiply $(8+2) * .15 * 1000 = 1500$ (the length of the buffer column). If enough memory is available, we would then dimension DIM W\$(25)[1500] (The absolute minimum safe size would be W\$(25)[1000]. Using this scheme. as we read records in from disk the sort fields starting with "A" would be assigned to the buffer column in W(0), the names beginning with "C" would be assigned to W(2), etc. The SORT command would be used to place each buffer column in proper order.

For those who use keyed-random access searching, this method is made to order for key insertion. After reading the relevant key sectors, any new keys are added to the bottom of the buffer, lightning sorted and put away. This method can be applied to the sequential arrangement of any type of data, so long as the weight of the ASCII characters used follows the desired arrangement of the output list. Combinations of characters are arranged by their binary weight; let your imagination be your guide from here.

HIDDEN POWER CORRECTION by Don Person

I must point out an error that I made in the keyword listings in "Hidden Power" (NEWS80S Issue #8). In dissecting the code for SCOPY, I made a MISTAKE. To set the record straight, I will give you the correct explanation.

SCOPY string1\$, string2\$ The value of string one is transferred to string 2. The transfer is made in 8 byte blocks, instead of byte by byte under ordinary system control. It greatly speeds up the process of transferring long strings from one to another. The program exploits the surfeit of 8 byte registers in the Series 80 cpu, and a benchmark test on string transfers could quickly persuade you of its usefulness too.

SCOPY ORIGINAL\$, DUPLICATES\$

Engineering Software

Entek Scientific Corporation offers standard software packages for use with HP desktop computers (Series 80 and 9800 Series) and most commercially available single and dual channel spectrum analyzers.

Application Packages Available For:

- | | |
|--|---|
| <input type="checkbox"/> Modal Analysis Studies | <input type="checkbox"/> Graphics Display and Automatic Mesh Generation |
| <input type="checkbox"/> Sound Intensity Measurements | <input type="checkbox"/> Machinery Health Monitoring |
| <input type="checkbox"/> Waterfall Mapping of Spectral Data | <input type="checkbox"/> General Purpose Analyzer Control and Enhancement |
| <input type="checkbox"/> Multi-plane Balancing of Rotating Systems | |

Custom designed software/hardware systems are quoted on request.

Call or write today for a copy of our DEMO PROGRAM and/or brochures.



ENTEK SCIENTIFIC CORPORATION
4480 Lake Forest Dr., Cincinnati, OH 45242
(513) 583-7500 Tlx. 241756

Change your weight . . .

Chubby Checker

Change your image . . .

Execu/Speak

Pocket more change . . .

Financialvision

Tape or 3 5 Disk \$95 each
Health & Habitation, Inc
Five Pathfinder Drive
Sumter, S.C. 29150

See H-P Software Catalog
or drop us a note for details

PLAN & MANAGE YOUR PROJECTS

HP86 *** CPERT *** HP87

- o SUCCESSFUL PROJECTS ARE WELL PLANNED AND TRACKED DURING EXECUTION TO MEET DEADLINES AND COST TARGETS. CPERT GIVES YOU THE POWER TO FORECAST, CONTROL AND TRACK THE PROGRESS OR LACK OF PROGRESS OF ANY PROJECT LARGE OR SMALL. THE BEST OF CPM AND PERT ARE COMBINED WITH ADDED PROPRIETARY FEATURES TO GIVE YOU THE EXTRA POWER AND CAPABILITY OF CPERT.
 - o CALCULATES CRITICAL PATH FUNCTIONS PLUS THE TIME, PEOPLE, LABOR AND OTHER COSTS OF EACH TASK. DATA CAN BE DISPLAYED OR PRINTED. GANTT CHART PLOTTED ON CRT OR PLOTTER. START AND COMPLETION STATUS OF EACH TASK IS MARKED ON PLOT.
 - o TIME BASE CAN BE IN HOURS, DAYS, WEEKS OR MONTHS. HOURS/DAY AND DAYS/WEEK CAN BE SELECTED AND ALL TIME BASE ELEMENTS CAN BE CHANGED AT WILL. EDITING OF ALL DATA INPUTS FOR CHANGES AND/OR 'WHAT IF'S' IS PROVIDED.
 - o CPERT CAN BE RUN AS A PURE CPM PROGRAM, PERT PROGRAM OR AS 'CPERT'.
 - o ERROR ESTIMATES ARE CALCULATED FOR EACH TIME AND COST ELEMENT IN EACH TASK, THE CRITICAL PATH AND THE TOTAL PROJECT. NOT ONLY IS THE TOTAL SLACK TIME OF EACH TASK CALCULATED BUT ALSO THE FREE SLACK TIME WHICH DETERMINES IMPACT ON EARLY START TIMES OF SUCCESSOR TASKS - A FACTOR IGNORED BY MOST PROJECT MANAGEMENT PROGRAMS - BUT CAN HAVE DRAMATIC IMPACT ON THE SMOOTH RUNNING OF A PROJECT OR PROGRAM. PROGRAM IS MENU DRIVEN WITH NO CYRIFIC COMMANDS.
 - o CPERT GIVES YOU FULL DETAILS OF ALL ASPECTS OF A PROJECT. YOU MAY DISPLAY OR PRINT AND PLOT ON THE CRT OR PLOTTER THE NUMBER OF PEOPLE, LABOR COST, NON LABOR COST OR TOTAL COST OF THE CRITICAL PATH, NON CRITICAL PATH OR TOTAL PROJECT TASKS. YOU MAY SELECT ANY TIME SEGMENTS OF THE PROJECT TO PLOT.
 - o PLOTS DONE ON THE CRT CAN BE COPIED TO THE PRINTER FOR REPORTS IF DESIRED.
 - o A DETAILED STATUS OR SUMMARY REPORT WITH CURVES (EVEN INCLUDING A COVER) CAN BE GENERATED IN MINUTES AND UPDATING STATUS IS SIMPLE AND FAST.
- DOES NOT REQUIRE CP/M PROGRAM MODULE
RUNS IN NATIVE BASIC PLUS SUPPLIED BINARIES
- o BASE PROGRAM REQUIRES 180KB TOTAL SYSTEM RAM AND HANDLES 225 TASKS. LARGER SYSTEMS AVAILABLE ON SPECIAL ORDER. BY USING SUB PROJECTS, BASE SYSTEM CAN HANDLE UP TO 50,825 TASKS !

\$175.00 PLUS \$3.00 SHIPPING & HANDLING

CPERT HELPS YOU PRODUCE THE ULTIMATE PRODUCT -- PROFIT !!

*** **

OTHER BUSINESS PROGRAMS FROM COLLINS & ASSOCIATES

- PLOT IT** - PLOT MULTIPLE LINE CURVES AND LINEAR REGRESSION CURVE FITS ON CRT OR PLOTTER USER RATED AS MOST POWERFUL LINE PLOTTING PROGRAM AVAILABLE FOR HP86/87.
- FILE IT** - VERSATILE DATA FILE MANAGEMENT SYSTEM. 86,000 CHARACTER DATA FILE. VERTICAL OR HORIZONTAL OUTPUT FORMAT. STATISTICS, ARITHMETIC, SEARCH, SORT, ETC.
- LC DATA** - SOPHISTICATED LEARNING CURVE PROGRAM FOR CALCULATION AND PLOTTING ON CRT OR PLOTTER. CALCULATES FROM ACTUAL PRODUCTION TIMES AND/OR COSTS.

CALL OR WRITE FOR DETAILS

ACCEPTED & LISTED UNDER THE HEWLETT PACKARD HP PLUS PROGRAM

COLLINS & ASSOCIATES, INC.

187 FLYING MIST ISLE · FOSTER CITY, CA 94404 (415)571-6991

THE HP150 TOUCH SCREEN COMPUTER by Dale Flanagan

Wait a minute. That's the HP150 personal computer? Why is NEWS80S reviewing a computer that's part of the Series 100 line? Although our world is the Series 80, it would be incredibly myopic of us not to cover what could become one of the most significant new products to ever come from Hewlett-Packard. The HP150 represents a radical change in HP product planning, positioning and marketing, and it therefore is much more significant than a new box of glass, plastic and metal.

What makes it special? Well, to start with the obvious things, the HP150 popularizes touch screen technology and makes it affordable on a personal computer. Touch screen is not new, but till now it's only been found on very expensive monitors designed for special purposes. On the HP150, a grid of invisible lights criss-cross the screen, and any object breaking this grid can transmit back to the computer the location of the "touch". Hewlett-Packard went to considerable trouble to make sure this system was easy to manufacture and up to usual HP standards of durability. The result is a method of control that makes a computer "mouse", the device used on the highly touted Apple Lisa computer, very unnatural and space consuming.

The resolution on the HP150 touch screen is 27 lines by 40 columns. This allows you to specify any point on a text screen to within two characters, and this resolution is quite practical for use with programs like Visicalc or even a word processor. When the grid of light beams is broken, a program can receive the location of where the user is pointing to as a sequence of escape codes. One area where this is especially useful is at the bottom of the screen, where highlighted "key label" boxes can be pointed to, instead of using separate function keys, as we have with the Series 80.

The monitor case also doubles as the computer case, and all the electronics for the CPU, memory and interfaces are packed into a box roughly the size of the Series 80 12" monitor (although the actual screen size on the HP150 is 9"). The computer comes with an Intel 8088 computer chip and 256K of memory. The memory can be expanded to 640K. Although big memory computers are not new to Series 80 owners, the use of an "off the shelf" computer chip is. The Intel 8088 is a 16 bit chip that uses an 8 bit bus. It's the same chip used by the IBM Personal Computer, and HP has followed IBM's lead in other key areas by adopting MS-DOS and Microsoft Basic as the standard operating system and language for the HP150.

MS-DOS is the disk operating system marketed by Microsoft, and "blessed" by IBM as the standard operating system for use on their personal computer. Because of the huge marketing clout of IBM, MS-DOS has become the most popular operating system for 16 bit computers. Microsoft Basic is by far the most popular Basic available, too, and the adoption of both these Microsoft products as "standard" should make a great deal of current IBM PC software available on the HP150 with relatively easy conversions.

At introduction time, HP is offering several packages that cover a variety of areas. Visicalc, Wordstar, Condor Database system, HP communications programs, a simple HP word processor, a graphics package and a neat, but simple, HP cardfile program cover most of the core activities HP150 users will be interested in. Several third party pieces of software, including Ashton Tate's dBase and the popular PFS database, will also be introduced for the HP150.

One disappointment at the initial software offerings is the fact that only Microsoft interpreted Basic is available for writing your own applications. Microsoft Basic is a good basic, but it is, in my opinion, clearly inferior to the HP Basic offered on the Series 80 (especially if you have the Advanced Programmer's ROM). Microsoft Basic doesn't allow named subroutines, it doesn't allow multi-line functions, and it can be relatively slow in execution speed. The Microsoft Basic Compiler, which solves this last problem, at least, isn't being offered by HP as part of the initial offering.

Although Microsoft Basic makes a ton of programs available to the programmer (versions of it are available on Radio Shack, Apple, IBM and almost anyone else you care to mention!), over the past few years I've used both Microsoft Basic and HP Basic, and the HP Basic is much faster for custom program development.

HP has talked about other language choices being available "soon", but no concrete information about this has been offered except for word that HP is working on its own version of HP Pascal for the HP150. This Pascal is supposed to be a subset of the Pascal used on the HP3000, which is an excellent implementation of Pascal with much needed file and I/O extensions.

The HP150 uses the same 3.5" disk drives that Series 80 owners have come to know and (at least in my case) love. This means that the HP150 owner can't just buy a piece of IBM PC software, which comes on a 5.25" diskette, and pop it into his or her HP. The small drives do, however, help give the HP150 an extremely small footprint of only 2.1 square feet, because the monitor/computer unit sits on top of the disk drive. As an extra cost option, HP will also sell you a tilt and swivel unit that fits between the disk drive and the monitor, so you can position the screen exactly as you like it. For the ultimate in compactness, an optional thermal printer will fit into a recessed area in the top of the monitor, giving you a complete desktop computer unit in an amazingly compact space.

With the HP150, the 3.5" drives hold only 256K of data, instead of the Series 80's 270K. There was considerable talk about a double sided 3.5" drive being available soon that would have 500K of storage, but no one could tell me if this double sided storage could also be used on a Series 80. At the HP150 press introduction a representative from the Greely, CO, disk division also mentioned a "high density" 3.5" drive they're working on that will hold one megabyte of data, but once again no concrete information was forthcoming about when these marvels would be let out of the lab and into consumer's hands. Frustrating.

For hard disk drives, HP offers a 5 megabyte and a 15 megabyte drive. The 10 megabyte drive available for the Series 80 will not be offered for the HP150, and the Series 80 will not be able to use the 15 megabyte drive (10mb is the current maximum). If all this doesn't have you confused yet, then the pricing of the 5mb and 15mb drives should give you some pause. With a single 3.5" microfloppy the 5mb drive is \$3040, and the 15mb drive with a single 3.5" microfloppy is \$3650. For an additional \$610 to triple the capacity of the hard disk, I can't imagine why anyone with a need for large capacity storage would buy the 5mb drive. Also curious is the pricing of the drive without the microfloppy. The 15mb drive is \$3345 without the floppy, but a single HP9121S 3.5" drive is still priced at \$900 by HP. This implies that the actual 3.5" drive unit is worth about \$305 (which sounds about right), and that we're paying a whopping \$595 for the case, power supply and HPIB electronics! Maybe the mile high air of Colorado has influenced these prices.

I'm harping on these pricing inconsistencies because a great point was made of the new marketing approach and pricing strategy taken with the HP150. In fact, the HP150 unit with dual 3.5" drives, 256K of memory, HP-IB interface, two RS232 ports, training software, and touch screen is a very competitive \$3995. This is less than an IBM PC with similar features (and without the touch screen, too), and I think the HP150 is a superior and faster computer. That's saying an awful lot, because the IBM PC is no slouch.

Now comes the big question that most Series 80 owners will wonder about: To heck with the IBM. . . is the HP150 better than the HP86B? To this question we give a fearless and unequivocal answer of "That depends. . ." If you're new to computing or if you're going to use your computer in a business environment, then the HP150 is probably a better machine. If you're going to do a lot of custom programming in Basic, or if you're going to use your computer in a technical or engineering environment, then I think the HP-86 is better (past issues of NEWS80S have talked about HP Basic and things like matrix ROMS, I/O ROMS, and Binary Coded Decimal computations used by the Series 80, which are why I think the Series 80 is superior in these applications).

HP apparently has the good sense to agree with me on this(!), because at the HP150 press introduction they said that they viewed the HP150 as their business and professional personal computer, and that the Series 80 was their entry level technical and engineering personal computer (I suppose the Series 200 is their "advanced" technical computer). Despite the minor gripes expressed in this article, I think the HP150 now gives Hewlett-Packard a dynamite 1-2 punch to go with their Series 80, and a legitimate contender for top PC honors with the IBM PC.

CENTRIFUGAL PUMP PAC
STEAM PAC
MECHANICAL SEALS PAC

for the HP 85/86/87

Pumps: Develops an approximate pump selection while checking for problem areas. Redraws pump curves for changes in rpm, impeller diameter, specific gravity and viscosity. \$295.00

Steam: Calculate properties of steam. Evaluate the Rankine Cycle. Determine theoretical steam rate. Approximate a typical steam turbine selection and steam rate. (HP-86/87 only) \$295.00

Seals: Calculate balance ratio, heat generation, temperature distribution, leakage, PV, flush rate and convective film coefficient. \$275.00

Gordon S. Duck
5801 Parkhaven Drive
Baton Rouge, LA 70816
(504) 293-6581

PACK'R for HP-85

PACK'R is a fantastic utility that takes an HP-85 Basic program and removes all remark statements and combines as many statements per line as possible.

The result? A more compact, faster running program that has more data space and that takes up less space on disk or tape.

The exact amount of program compression and program speed-up will vary from program to program, but in test we've had size reductions of 25% and speed improvements of 7%.

Basic program and binary program on HP-85 data tape, complete with instructions, only \$39.95!

RAMBLING by Dale Flanagan

THE INTRODUCTION OF THE NEW HP150 brings speculation about what other moves HP has up its sleeve. The people at Corvallis have been unusually close mouthed, so we haven't had too much help in discovering what else we may expect in the way of new products. One persistent rumour is that Corvallis will come out with a new portable or briefcase computer next year.

Actually, it would make sense for HP to come out with both a portable, briefcase style computer as well as a smaller lap computer, because both segments of the market are extremely hot right now. In the portable category, the Compaq and the KayPro are both doing well, and in lap computers the Radio Shack TRS-100 is currently cutting a wide swath. With these computers, bundling software with the basic unit and sensitivity to pricing are two key elements if the unit is to be a success. I also feel that the display size and keyboard size are two critical factors with this type of unit.

It's no secret that HP intends to become a much bigger force in personal computing than they have been up till now, but it's also comforting to know that they say they intend to do this without sacrificing traditional HP virtues like quality construction and service.

If new products come out of Corvallis, I hope they make them compatible with other HP products. It makes sense for Corvallis to use MS-DOS, like the HP150 (although for sentimental and purely selfish reasons I'd like them to be compatible with the Series 80), but your guess is as good as mine as to what they really will use. Also for sentimental reasons, I hope that the new products are called "Series 80" computers, too.

Most rumours peg an introduction date of the first or second quarter of 1984 for new products out of Corvallis.

SPEAKING OF RUMOURS, I've noted that our publication has been the subject of speculation lately by members of the small Series 80 community. Although nothing has been finalized as I write this, we have been exploring ways to radically upgrade the format and timeliness of NEWS80S

You, our readers, have given us tremendous support over the past two years. When we started NEWS80S we set a circulation goal of a few of hundred, and instead we've ended up with a few thousand readers! This makes the publication, as you see it now, very viable and healthy. This circulation, however, is not enough to support typesetting, photographs, color and artwork. To take this next step, our circulation would have to increase by quite a few readers, and we would need the help of an experienced production and art staff. One way to do both these things is to work with a publisher interested in serving the needs of the Hewlett-Packard computer community, and we have been talking with just such a publisher.

To dispel the most persistent rumour we've heard about us, we have not been talking to an old east-coast publishing house that wants to venture into computer periodicals. Instead, we've been talking to a west-coast magazine publisher that has been very successful in the IBM PC market, and who is already in the process of expansion. We felt that we wanted to deal with people already experienced with computer magazine publishing who would be flexible enough to agree to (1) making sure that the interest of our current readers are protected, (2) allowing NEWS80S to maintain its own identity and flavor, and (3) interested in ending up with a quality publication that accurately reflects the interest and needs of the Series 80 owner.

Stay tuned, and we'll tell you if we were able to achieve those goals!

YOU MAY THINK I JUST FELL OFF OF A CABBAGE TRUCK, but I have an awful confession to make: until recently I didn't own an HP calculator! I always purchased calculators that didn't use RPN notation, but my Series 80 Assembly language work has taught me that RPN is really not too bad, so I recently picked up an HP-41. It's fun.

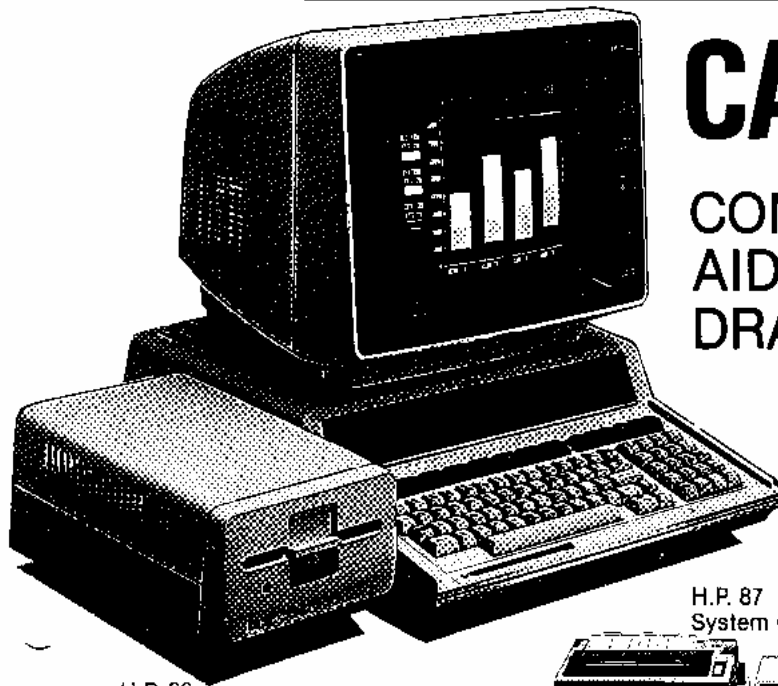
While I don't classify RPN on an HP calculator as being in the same league as Pac-Man, I'll have to admit that I've actually derived quite a bit of amusement playing with the 41. Much of this enjoyment came from reading HP's publication, *Keynotes*, which was edited by Henry Horn. I called Henry to find out about back issues of *Keynotes*, and was very sad to learn that he was retiring from HP, and that no other editor for *Keynotes* has been hired. Henry sent me back issue information, and I was later surprised to receive a long letter from Henry and a special copy of his last issue of *Keynotes*. It struck me that Henry Horn must be an exceptionally thoughtful person, because amidst all the bustle of his retirement process, he spent the time to respond to an information request from a total stranger. I'm sure HP and HP calculator owners will miss him.

SPEAKING OF HP PUBLICATIONS, some of our readers may wonder what happened to the very fine Basic Exchange journal that THP used to put out for the Series 80. It's been over a year since the last issue, and while no one will tell me that Basic Exchange is dead, no one can tell me when the next issue will appear, either. When Curt Adams left the editorship of Basic Exchange to seek his fame and fortune in programming, HP didn't hire another editor. Since I figure you can't have a publication without an editor, I also figure that Basic Exchange must be dead.

A NEW BINARY FOR PROGRAMMERS! by Don Person

Has this ever happened to you? You have this fine chunk of program with a few un-resolved line branches. (You know, you entered GOSUB DINGO and your linelabel read DINGA: right?). You would like to renumber it to open up some more space, during what we hope is the top down refinement portion of the show. One problem, though, is that all you get is an error message for missing line references. Your AP rom commands, like SCAN, XREFL, XREFV and RENUM aren't of any use either. Till now there were only two ways out. First go through your code till your eyes are squinting, hoping that you'll luck out. The other involves setting up a BASIC program that will help you do it by evaluating the strings in a GET/SAVE file, checking for branches by their position with respect to other keywords.

RENUMB ends all that forever. When an unresolved branch is encountered in a renumbering operation, the system still pauses and returns the MISSING line message. If you then type WHAT LINE , the binary returns a number, which is of course the line number of the line containing the conflict. If you've ever been through any of this, I need not say more. I don't know the price, but if it's priced at less than \$1000.00, it's worth it. After trying it out on several programs myself, I can't find any flaws. I would personally guarantee that this is one binary you will never regret buying! (Binary programs are available from the Series 80 User's Library, 1010 N.E. Circle, Corvallis, OR 97330).

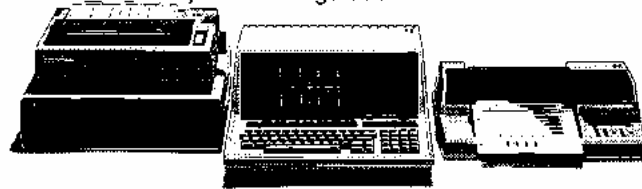


H.P. 86
Personal Computer

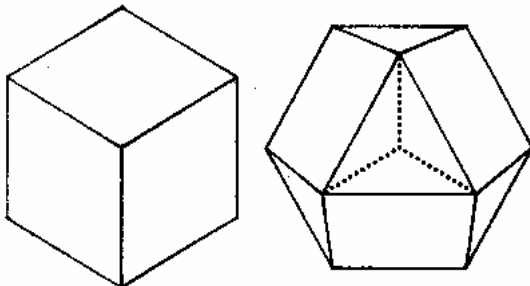
CADD/86-87

COMPUTER AIDED DRAFTING

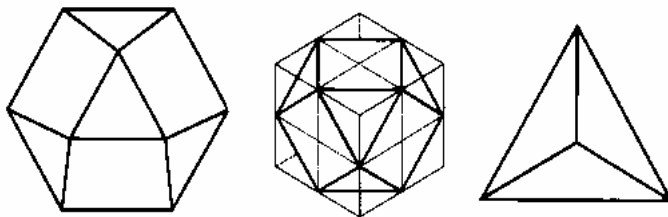
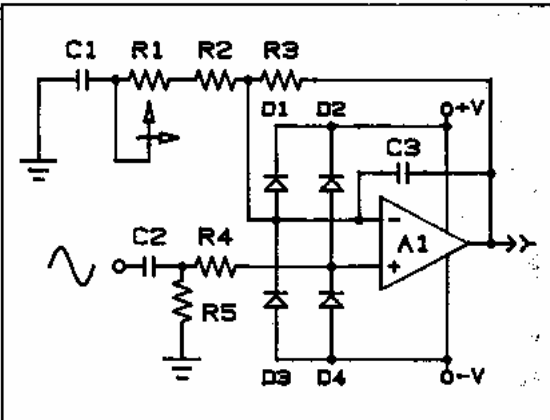
H.P. 87
System Configuration



CADD/86-87 2D DRAFTING



COMPUTER DRAFTING BY TENSEGRITY INC.



CADD/86-87 IS GENERAL PURPOSE...
USEFUL FOR MANY
DESIGN / DRAFTING APPLICATIONS

HP logo This product is part of HP FOCUS - A program for locating user-aided software. It is a guide for HP PLUS as determined by references from consultants and users. HP FOCUS software is provided by independent software suppliers for operation on HP computer systems. The supplier is solely responsible for its software and support services. HP is not the manufacturer or co-developer of such software or support. HP claims no warranty and will not be liable for any damage, howsoever, expressed or implied, with respect to this software. Distribution of this product or information concerning this product does not constitute an endorsement of the product, the supplier, or support services. The customer is responsible for selection of the software it purchases.

TENSEGRITY, INC.
2424 ADDISON ST. CHICAGO, IL 60618
312-935-9714

(Advertisement)

SUPER SNOOP'R

A Series 80 Disk Utility Program written by David Efron, D & E Systems

SUPER SNOOP'R is the most powerful Series 80 disk utility ever offered. It's designed to be used as an educational package and a program development tool. It allows the HP—86/87 owner to look at, study, and change any sector on a standard HP Series 80 diskette.

You can view your diskettes and see how data and programs are stored. You can see the differences between Basic programs, Binary programs, and data files as you learn about the fascinating inner workings of your Series 80 disk system.

For program development, SUPER SNOOP'R can help you in many powerful ways. For instance, if you're having problems with the LINPUT command when trying to use the binary program UTIL/1 with the Advanced Programming ROM, SUPER SNOOP'R will let you change the UTIL/1 "LINPUT" command to "LINGET", to avoid this conflict. SUPER SNOOP'R allows you to modify any Binary keyword quickly and easily. In fact, SUPER SNOOP'R will allow you to view and modify any single byte on a diskette!

Operation of SUPER SNOOP'R is controlled by the function keys on your HP—86 or 87. You can view files and programs by name, or select any sector on the disk by number. Your computer screen displays the sector to you as ASCII characters and as octal or decimal numbers. SUPER SNOOP'R keeps you informed of the file name and sector number you're looking at, and you can step forwards or backwards through a file sector by sector, or jump directly to the sector you're interested in. With the touch of a key the contents of a sector will be sent to your printer. If you want to change a sector, these changes can be entered as decimal numbers, octal numbers, or ASCII characters.

Because of the power this utility puts into the user's hands, it is possible for a careless user to damage the data or programs on a diskette, making them inaccessible or inoperable. For this reason, we recommend that SUPER SNOOP'R only be used on diskette copies, never on originals. SUPER SNOOP'R is very easy to use and comes with a comprehensive manual, but we don't recommend it for beginners. We do recommend it for advanced programmers and anyone interested in learning about the details of the Series 80 disk storage system. For this purpose, SNOOP'R truly is SUPER!

INTRODUCTORY PRICE Until February 1, 1984, we will be offering SUPER SNOOP'R at the introductory price of \$ 99.95 After February 1st, the price of SUPER SNOOP'R is \$ 119.95. Add \$2.00 shipping and 6.5% CA sales tax, if applicable.

ANOTHER SERIES 80 PROFESSIONAL SOFTWARE PRODUCT by Joseki Computer Corporation.

P.O. Box 1329, Redondo Beach, CA 90278

The HP Microcomputer Journal

SPECIAL IN-DEPTH REVIEW OF THE HP-86/87 ASSEMBLER ROM

PLUS

ASSEMBLER BREAK-POINT SUBROUTINE - COMPLETE LISTING INSIDE!

```

9000 BRKPT   SAD                ! Save before we do anything
9005        PUMD R70,+R12      ! Save the 70's, 60's and 20's
9010        PUMD R60,+R12      ! because we have to use them
9015        LDM R70,R20
9020        PUMD R70,+R12
9025        BCD
9030        CLB R22            ! Shift E into a zero'd byte
9035        ELB R22            ! (four bits at once in BCD mode)
9040        BIN
9045        PUBD R22,+R6       ! Put E onto stack
9050        PUMD R70,+R6       ! Save the 70's again, so we can use them
9055        LDM R22,=BRKTABLE  ! We need R22-23; they're clobbered,
9060        STM R22,R75        ! but we save them above.
9065        CLB R77            ! Set up for PTR2 stores
9070        ADMD R75,=BINTAB   ! (The absolute address of our save area)
9075        STMD R75,=PTR2
9080        POMD R70,-R6       ! Now . . . Save the 70's in the save area
9085        STMI R70,=PTR2-
9090        STMI R60,=PTR2-    ! And the 60's
9095        STMI R50,=PTR2-
9100        STMI R40,=PTR2-    ! The 50's and 40's
9105        LDM R70,R30
9110        STMI R70,=PTR2-    ! For the 30's on down to the 0's, first
9115        POMD R70,-R12      ! move them to the 70's, EIGHT registers
9120        PUMD R70,+R12      ! at a time.

```

PLUS

THE HP150 - WHY ARE WE REVIEWING THIS COMPUTER?!!?

SORTING IT OUT - Practical advice on sorting

news80s

The HP Microcomputer Journal

*** ANNOUNCEMENT *** ANNOUNCEMENT *** ANNOUNCEMENT ****

IS THIS OUR (GASP!) LAST ISSUE?!?!?

THIS IS THE LAST ISSUE OF NEWS80S IN OUR CURRENT FORMAT. BY THE TIME YOU READ THIS, WE SHOULD HAVE COMPLETED PLANS TO SELL NEWS80S TO A NEW PUBLICATION, HIGH PERFORMANCE COMPUTING. I WILL BE THE EDITOR OF THIS NEW PUBLICATION.

NEWS80S WILL REMAIN A DISCRETE SECTION OF HIGH PERFORMANCE COMPUTING. WE INTEND TO MAKE SURE THAT THIS SERIES 80 SECTION WILL CONTINUE TO SERVE YOUR INFORMATION NEEDS WITH GREATER TIMELINESS, DEPTH AND STYLE THAN WE COULD WITH OUR CURRENT NEWS80S RESOURCES.

HIGH PERFORMANCE COMPUTING WILL BE A COMPLETELY TYPE-SET, FULL COLOR MAGAZINE. WE'LL BE COVERING THE NEW HP 150 COMPUTER AND THE SERIES 80S COMPUTERS EXCLUSIVELY. WE WON'T COVER CALCULATORS AND WE WON'T COVER HIGHER PRICED HEWLETT-PACKARD MACHINES. WE'LL CONCENTRATE ON THE HP PERSONAL COMPUTERS USED BY BUSINESS-PEOPLE AND PROFESSIONALS, AND BY NOT SPREADING OURSELVES TOO THIN, WE THINK WE'LL BE ABLE TO CONTINUE TO BRING YOU INFORMATION OF VITAL INTEREST TO OWNERS OF SERIES 80 COMPUTERS. ALTHOUGH WE'LL BE HEWLETT-PACKARD SPECIALIST, I THINK IT'S WORTH MENTIONING THAT HIGH PERFORMANCE COMPUTING WILL BE AN INDEPENDENT PUBLICATION, SO WE WON'T BE AFRAID TO TELL YOU IF A PIECE OF HARDWARE OR SOFTWARE DOESN'T MEASURE UP, NO MATTER WHO'S NAME IS ON THE BOX.

HIGH PERFORMANCE COMPUTING WILL BE PUBLISHED 6 TIMES PER YEAR. A 6 ISSUE SUBSCRIPTION IS \$18. YOU WILL RECEIVE A ONE-FOR-ONE CONVERSION OF YOUR REMAINING NEWS80S SUBSCRIPTION TO HIGH PERFORMANCE COMPUTING. IN ADDITION, WE WILL EXTEND YOUR SUBSCRIPTION BY AN EXTRA ISSUE, JUST TO SAY THANK-YOU FOR YOUR PAST SUPPORT. WE'RE VERY EXCITED BY THIS DEVELOPMENT, WHICH OCCURRED RIGHT BEFORE PRESS-TIME (SEE THE RUMOUR IN THE RAMBLING COLUMN), AND WE HOPE YOU'LL BE PLEASED BY OUR NEW FORMAT, NAME AND EXPANDED COVERAGE. LOOK FOR THE FIRST ISSUE OF HIGH PERFORMANCE COMPUTING IN YOUR MAILBOX IN MARCH OF 1984.

THANK YOU FOR YOUR CONTINUING SUPPORT,



DALE FLANAGAN
EDITOR

n@w@s80s

The Microcomputer Journal For HP Series 80

**News80s ran for 12 issues between 1982 and 1984
(#1, #2, Special Issue and #3 thru #11).**

**It was an independent newsletter edited by Dale Flanagan for:
HP-83, HP-85, HP86 and HP87 Personal Computer users.**

These are used with the permission of Dale Flanagan, who retains the copyright.

Scanned and converted by M. A. Cragg

NETWORKING™

-Limited time
introductory offer-

Only \$250

NETWORKING lets you network with other computers over telephone lines. It is a system of programs that combines text editing, file management, and communications. Your text files can be easily manipulated by the various elements of the pac, increasing the utility and power of your data.

The text editor and file manager are optimized for electronic communications, so they're easy to use. They aren't intended for high-volume secretarial word processing or numerical database manipulations so there are no embedded formatting codes or esoteric database management techniques to learn. If you can run any other application program, you can run NETWORKING or your money back.

The one feature that no other communications program offers is a communications language that permits customizing your system for your personal selection of databases and operating procedures. Your computer can operate automatically, even in your absence at night when the rates are considerably lower, talking with other computers and actually making decisions about what to do based on the host computer's responses.

Imagine your computer waking up at 2:00 a.m. and sending ten telexes to Europe via Western Union's *EasyLink* or calling The Source to send a message to a dozen field offices, read all your electronic mail, and check the UPI wire for the latest stories on the space shuttle. Then at 3:00 your machine logs onto Dialog's *Knowledge Index* to retrieve abstracts of specific research reports published since the last time you checked. Similarly, you can keep up to date on the business market environment by periodically searching the text of *The Wall Street Journal* using the Dow Jones News/Retrieval Service's free text search capability. Anything that can be done from the keyboard can be programmed to be done again, automatically.

NETWORKING™ is a trademark of Gaia Communications

-Available only from-

Gaia

The user's manual is available for \$35.00, which can be deducted from a later purchase. Call or write to order or for more information.

Gaia Communications

Rt. 2 Box 321C

Corvallis, OR 97333

(503) 929 4055