CROSSTALK Journal of Hewlett-Packard Technical Computer User Groups August, 1982

HP AUSTRALIA MOVES INTO SOFTWARE BUSINESS

After months of investigation and planning, Hewlett-Packard has formally established its first Australian manufacturing facility. Focusing entirely on application software and tools, the Australian Software Centre (ASC) will bring to HP customers quality hardware and software solutions from a single vendor.

John Cromie, formerly the Southern Commercial District Systems Engineering Manager, took up the challenging new role of Software Centre Manager on February 1st.

"While the ASC will be producing software products for the entire HP computational product range, we anticipate that a number of our early products will be directed towards the technical computer range," said John Cromie when interviewed by Crosstalk.

"HP has produced some first-class application software for the Manufacturing industry, but we need to broaden our offering to cover other key areas, such as Computer-aided Drafting, Job Shopping and Process Control".

The ASC also has responsibility for the administration of the HP PLUS program, through which quality third party software is brought to the attention of prospective users through HP's sales force.

"The HP PLUS program has been developed to provide our customers with high quality solutions to their problems using HP hardware and software supplies. Under HP PLUS we will assist software suppliers who have a superior total package of software, support and documentation," said Cromie. "The common goal we share is customer satisfaction".

HP1000 Directions Presented

Ed Brumit of Hewlett-Packard's Data Systems Division recently travelled through Australia in conjunction with HP's Productivity '82 show. The HP1000 Users Groups in Melbourne, Sydney and Brisbane took this opportunity to have Ed present directions of the HP1000 computer family at their regular meetings.

The technical computer line continues to grow with the new A600 and A700 processors. The A600 is a microcomputer which features one million instructions per second and an input/output bandwidth of 4.3 megabytes per second. The A700 minicomputer is also a one million instruction per second machine and has an input/output bandwidth of 3.9 megabytes per second. This machine's main features include an acceleration processor (hardware floating point, SIS and VIS on one board) and user microprogrammability. Both computers use the DMA per channel architecture originally introduced on the L-Series and are available in a variety of packages. The operating system for the A-Series, RTE-A.1, supports up to 4 megabytes of main memory and is the first RTE to allow dynamic memory allocation. The new machines are aimed at OEM's and target automation applications, while the traditional HP1000's (E-Series and F-Series) will continue to be the processors for the scientific and heavy program development environments.

Data Systems Division will be introducing several new software products in 1982. Image/1000 will be enhanced to include data types for double precision real numbers and double word integers, transaction logging, a data base recovery

CONTENTS

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HP 1000 Directions presented (Cont.)

routine and data set locking capability. New software products will be introduced to help production managers to track and analyze product quality, engineers to set up and manage measurement and control systems, systems designers to build redundant computer systems, and programmers to write and debug programs more quickly and efficiently.

Current plans also call for introduction of another new HP1000 processor before the end of 1982. This will be the highest performance computer ever introduced by Hewlett-Packard. Ed said more about this machine in his presentation, but did not want it committed to print. (Maybe this will encourage those members who did not make it to this meeting to be more diligent in their future attendance).

You can see that projects are moving along very quickly at HP, but apparently not quickly enough. Hewlett-Packard is also putting a great deal of effort into the HP PLUS software program. This program has been built to seek out, qualify and document software packages which are commercially available for HP computers. Over fifty packages are currently listed which run on HP1000's. With the improved price/performance ratios on the new computers and more software solutions available from Hewlett-Packard, the HP PLUS program and even from the HP1000 International Users Group, electronic data processing is becoming practical for a whole new spectrum of users.

ADA

By Shirley Alexander
(02 699 7181)

I have recently been to the U.S.A. and spent some time investigating this new language. I hope to give you a brief introduction to Ada and mention features which may be of interest to HP1000 users.

The language is named after Ada Augusta, Countess of Lovelace, daughter of Lord Byron, and colleague of Charles Babbage. She is credited with being the world's first programmer.

Ada was developed at the instigation of the U.S. Department of Defence (DOD) who needed a standard structured language for the needs of Navy, Army, and Air Force. The DOD had very bad experiences with their embedded systems for weapons, missiles and satellite control. The main programming languages used for these applications are Jovial and CMS/2, while NASA has used its own language HAL/S for control of the Space Shuttle. The problems lie in the dissimilarities and bugs in the various versions of these languages (eg. the Navy has 16 different types of computers which run these applications). The DOD is now determined to have a standardized compiler, with as few bugs as possible, and to pay only once for the "front end" and then smaller sums for each "back end" which produces different machine languages for different target machines. The contract for validating ADA compilers has gone to Intermetrics which is using the ADA-ED translator from New York University to check its validation tests.

One of the important features of ADA is parallel programming. Since Ada is written for embedded systems, programs need to be able to deal with many activities that are happening at the same time. Ada recognises this and permits programs to be written using tasks. Tasks are concurrently executable modules, to allow for many activities which are happening at the same time, which usually occur in embedded systems controlling complicated experiments, weapons or engineering systems. Interaction between tasks is controlled by the ADA concept of rendezvous, in which two tasks coincide for a period of time, after which they resume their parallel execution.

Another feature of ADA which has been developed specially for control systems is the idea of exceptions. These provide a mechanism for the unusual termination of program units. These exceptions may be defined by the user, or pre-defined by the ADA systems such as INDEX-ERROR or OVERFLOW. Exceptions may be executed by means of the key word RAISE:

e.g.

```
IF DETERMINANT = O THEN
RAISE SINGULAR;
ENDIF;
```

The language is not without critics however, and there are widely divergent views about it. At one extreme, Prof. I.C. Pyle of the University of York says "Ada is a major advance in programming technology, bringing together the best ideas on the subject in a coherent way designed to meet the real needs of practical programmers. It is the first result of a substantial effort to identify the requirements for programming and satisfy them effectively".

On the other hand Dr. Herbert Grosh, past president of the A.C.M., calls ADA "the world's most repellant language".

With numerous firms jumping on the bandwagon to produce Ada Compilers (eg. Telesoft's compiler to run on the Motorola 6800) in the face of many criticisms, Ada is at the crossroads of becoming a widely accepted language or merely restricted to use by the U.S. Department of Defence.

LANGUAGE WORKSHOP

A one day language workshop will be held on Thursday the 30th September for anyone considering the use of PASCAL as a programming language. The strengths and limitations of the language will be discussed in comparison with other programming languages. Speakers will include Dr. Bob Pascoe, a leading authority on PASCAL, and lecturer in Computer Science at RMIT, as well as several HP Computer users who will talk about their experiences in using PASCAL in applications programming. The seminars should be of interest to HP1000, 3000 and desktop computer users. The venue will be Hewlett-Packard's, Blackburn office, between 9 am. and 5 pm. A fee in the range $25 to $40 will be charged to cover costs and a light lunch. For more information contact:

JOHN GWYTHER
President, HP1000 Users Group, Vic.
Tusc Computer Systems Pty. Ltd.
696 High Street, Kew East.
Vic. 3102.
Phone: (03) 859 9487.

Canberra Technical Users Group Alive and Well

After a period of inactivity, a technical users group is again active in the Canberra region.

Between 1975 and 1980, users of HP desktop computers held regular meetings to disseminate their knowledge and expertise. In November 1980 the interests of this group were broadened to include minis. After a couple of meetings early in 1981, the Users Group became dormant until 18 May of this year when it was reformed with the following aims :

Meetings to be bimonthly
Meetings to be held 8 pm. Tuesdays
Meeting venues to be varied between HP's Canberra office and other users' sites
Content of meetings to include details of all new HP products with demonstrations where possible, news of future releases, visiting speakers, users' software, application stories from local and other users.

The following office bearers were elected:

President: Bruce Wyatt Data Science (062) 80 4770.
Alternate President: Colin Griffiths Defence (062) 66 4184
Treasurer: Barbara Harrison Nat. Parks & Wildlife (062) 89 7919.

Members of the Technical Users Group include members of several consultant and processing companies as well as representatives from many Government departments and bodies (ANU, CSIRO, National Parks & Wildlife, ACT Forests, Communications, National Mapping, Defence (Navy, RAAF, Transport, Communications), Bureau of Mineral Resources). Equipment configurations and applications vary widely between users' sites.

The meeting of 20 July featured John Kean from Hewlett-Packard's Sydney office. John described the HPIL interface loop and indicated various configurations including the HP85 personal computer, 82905B printer, 3468A multimeter, HP41C calculator, 82161A cassette drive, 82162A thermal printer, and other HP41C modules. Likely future HPIL devices, interfaces, and applications were discussed in length.

Grant Spratt from HP Canberra described other new products released during the last two months. John and Grant demonstrated various products including the 16C and 41CV calculators, the HP85, HP87, and 9826 desktops, and various peripherals.

The next meeting is to be held at 8 pm. on Tuesday, 28 September 1982, at Data Science's office, 97 Wollongong Street, Fyshwick ACT. This meeting will feature an application example and demonstration by John Rees. John will describe a database and graphics system developed by Data Science for the mining and exploration industry. This will be followed by a demonstration of software packages on HP1000 and HP85 systems. All visitors will be welcome.

Bruce Wyatt
President
HP Technical Users Group — Canberra Region

Correspondence to: P.O. Box 507 Fyshwick. ACT. 2609

WHAT IS SRM?

I was prompted to answer this question, by way of this article, at the latest meeting of the H.P.D.C.U.G. In responding to a question from one of our members, Dennis Vetter, our guest speaker, was heard to ask "Does everyone here know about SRM?", and was met with deafening silence. In this article I shall attempt to bring all our readers up to date on this very question.

SRM stands for Shared Resource Management. The notion of sharing resources is a fact of life to any mini - or mainframe user. HP1000 users are quite accustomed to sharing disk space, printers and even the computer's CPU. However one of the problems faced with sharing, for example the CPU, is that the more users who share the CPU, the worse the performance that each user experiences. On the other hand, one of the big advantages with everyone being able to share a large hard disk, is that many users may be given access to common data and programs via group cartridges, differing access rights etc.

All this is very foreign, however, to the desktop user, who has maximum performance with an entirely dedicated system. The price we have to pay for this is two-fold.

(i) Because every user has his/her own system, the cost of buying a full complement of high quality peripherals, e.g. line printer and hard disk is too great. Hence we make do with lower-performance peripherals.

(ii) Because every user operates in his/her own environment, there is little, if any, opportunity to share current information on disk.

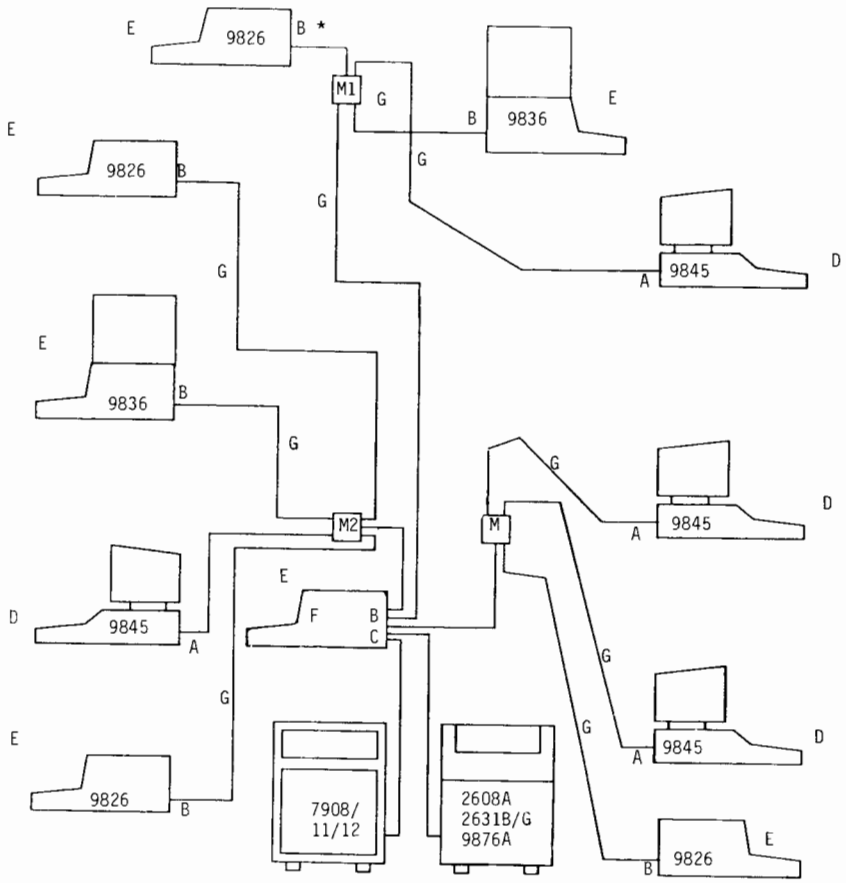
The Shared Resource Management system (SRM) gives us the best from both these worlds.

The SRM is, basically, a dedicated 9826 with its own operating system, resource management multiplexer, cables and interfaces, etc. This enables a 9826, 9836 or 9845 to be configured in a network which can share various resources. (See fig. 1). While it is controlling the system resources, the 9826 is dedicated to this task, under its compiled-Pascal operating system. It is however, a normal 9826 which could be booted in any of its 3 languages if it were not doing its SRM task. The 9826 includes a new disk interface, DMA card, and resource management interface.

The SRM system:

(i) configures the network at power-up.
(ii) performs multi-user file management.
(iii) defines peripherals attached.
(iv) sets up spooling to a printer.
(v) initialises the disk.
(vi) sets time and date.

The System



A Resource Management Interface for 9845B/C - pn 98029A
B Resource Management Interface for 9826A/B - pn 98629A
C Disc Interface - pn 98625A
D Resource Managment ROM for 9845B/C - pn 98419
E Non-scratchable Binary, soft ROM for 9826/36 - pn 09826-10020
F Shared Resource Management Operating System - 98261A Opt. 500
G Cables, available in 10m, 25m or 60m - 98029 Opt. 001, 002, 003, 004
M Resource Management Multiplexer - pn 98028A

The multiplexer links the individual desktops to the SRM manager, one multiplexer needed for every four desktops. The number of multiplexers is limited by the slots available for interfaces in the backplane of the SRM manager. The mux has one cable for both power and data and four ports for cables to transmit to other desktops on the network and the SRM manager. The multiplexer can be powered by any 9826/36 (not 9845) on the network. This enables a maximum end to end connection of 240 metres. Cabling provides 700 K bit serial data exchange rates in packet lengths up to 700 bytes. Polled access is used to avoid collisions.

The SRM manager will drive up to 4 hard disks in the CS-80 family (i.e. 7908, 7911, 7912). Printers include 2608A, 2631B/G and 9876A. Plotters? Not now, but .....

Of the advantages of SRM, the sharing of files is one of the most important. A new tree-structured directory allows files to be logically grouped. ASCII files can be read by any 9826/36 or 9845 on the network. A STOREd 9826/36 program can be accessed by any other 9826 and likewise a STOREd 9845 program can be accessed by any other 9845. File types allowed are DIR(ectories), DATA, PROGRAM, BDAT, BIN and ASCII.

(Note, **not** 'IMAGE' files — not now anyway). File names can be up to 16 characters with up to a 16 character password.

Having DIRectory file types enables a user to have his/her own private disk area which appears similar to the familiar floppy. A user can assign passwords allowing manager access, and/or read-only access, and/or write only access to selected users. For example only a user with manager capability can PROTECT, RENAME or PURGE a file.

Some minor changes to existing programs will have to be made to access the shared hard disk and enable printer spooling. A typical example of accessing the hard disk would be:

```
10  DIM Pathname $ [150], Filemane $ [34], MSUS $ [30]
          :
          :
200 Pathname $ = "SEO/PHIL-GREETHAM/SLIDES-DATA/"
210 Filename $ = "SLIDES-1"
220 Msus $ = ":REMOTE 5,1"
          :
          :
300 ASSIGN #1 TO Pathname $ & Filename $ & Msus $
310 PRINT #1; Data (*)
320 ASSIGN #1 to *
```

The pathname defines the path down through the tree-structured directory. Note that if this is omitted, the file access looks just like it used to. Filename is 34 characters to allow for a 16 character name, 16 character password and left and right carats (**< >**).

Printer spooling is not quite as simple as saying PRINTER IS ":REMOTE". It entails creating a file in the spool directory and outputting strings to it. After the file is closed the SRM takes control of it, locks it, prints it out to the printer and purges it.

Space does not permit me to go into details on performance. Briefly, for a disk access by a single user on the network, the performance is better than a 9826/36 using the internal floppy, and about the same as a 9845 using a dedicated 9885. About the worst performance that could be expected would be if all users on the network tried to simultaneously read or write different, large files. In this case the performance would be slowed by a factor of about 6. For non-disk intensive applications users will rarely make concurrent requests. Consequently the throughput will be much closer to the single-user performance than multi-user.

In a nutshell, SRM gives you all the power of a single-user system, whilst being able to share expensive peripherals and files with other users on the network.

Philip Greetham
Systems Engineer
Hewlett-Packard, Melbourne.

Dozens turned away from User's Group meeting

Dozens turned away — well that's what we thought could happen but unfortunately the notice of our meeting of the desktop computer users group at the Shoppingtown Hotel didn't reach most members in time for them to rearrange their busy schedules and as a result the meeting was only moderately well attended. The committee has taken this problem in hand and future meetings will be advertised well in advance.

While waiting for our guest speaker who was flying in from W.A. we discussed the first issue of CROSSTALK. Members praised the standard of production and requested more hardware and programming snippets and odd bits and pieces of information of all types.

Further discussion brought out that members thought that workshops on programming techniques, structured programming and the like would be of great value to them. The committee will look at the idea of organising workshops and will seek input from members on when and where they could be scheduled. (Call a committee member if you have any ideas — Ed.).

Our guest speaker Dennis Vetter, Manager of International Marketing and Support of the Desktop Computer Division arrived and plunged straight into it. He explained the Shared Resource Management system concept using desktop computers to improve the productivity of engineers and others in their working environment. He then went on to speak about two major product development programs at H.P., one based on the Motorola 68000 series of microprocessors and the other on H.P.'s own micro. Both programs show great promise for the future.

From hardware Dennis moved onto software, pointing out that H.P. intended to strengthen this area with more emphasis on contributions from 3rd parties. This produced an immediate reaction from his audience who quickly asked a lot of questions. At this juncture Bruce Miller took the floor to explain H.P.'s plans for software support in Australia. (See article elsewhere in this issue).

The meeting then broke up for drinks (kindly provided by H.P.) and informal discussions with our visitors.

Thanks to Dennis Vetter for squeezing our meeting into a busy itinerary.

B. T. O'Shannassy.

I/O ROM Uses

Functions provided on an Input/Out ROM give programmers very powerful control over input and output for HP desktop computers.

There are other uses, however, to which such functions can be directed. In this and other articles, I propose to present a few small routines and explanations that may prove handy in writing programs for desktops.

The Advanced Programming ROM for the HP 85 gives a total of 64 flags which can be set, cleared or tested at any time. The I/O ROM can provide this capability for as many flags as you wish to select. Using a string of 256 characters, for example, there are 2,048 flags available to you. Each bit of each character byte is treated as a flag and can be set, cleared or tested using three small multiple line function routines. The listing of these is given below:

```
10 DIM F$[256]
20 INTEGER F6(7),F7(7),F8,F9,Z9
30 FOR I=0 TO 7
40 F7(I)=1*2^I
50 F6(I)=255-1*2^I
60 NEXT I
70 F$=" "
80 FOR I=1 TO 4
90 F$=F$&"<<<<<<<<<<<<<<<<<<<<<<<<
   <<<<<<<<<<<<<<<<<<<<<<<<<<<<
   <<<<<<<<<<<<<<<"
100 NEXT I
110 GOTO 500

120 DEF FNS(Z9)
130 F8=(Z9-1)\8+1
140 F9=-(Z9-8*F8)
150 F$[F8,F8]=CHR$(BINIOR(F7(F9)
    ,NUM(F$[F8,F8])))
160 FNS=Z9
170 FN END

180 DEF FNC(Z9)
190 F8=(Z9-1)\8+1
200 F9=-(Z9-8*F8)
210 F$[F8,F8]=CHR$(BINAND(F6(F9)
    ,NUM(F$[F8,F8])))
220 FNC=Z9
230 FN END

240 DEF FNF(Z9)
250 F8=(Z9-1)\8+1
260 F9=-(Z9-8*F8)
270 FNF=BIT(NUM(F$[F8,F8]),F9)
280 FN END

500 DISP "CLEAR,SET OR TEST";
510 INPUT A$
520 IF A$="CLEAR" THEN 600
530 IF A$="SET" THEN 700
540 IF A$="TEST" THEN 800
550 BEEP ● GOTO 500
```

```
600 DISP "CLEAR WHICH FLAG (1-2048)";
610 INPUT Z9
620 IF Z9<1 OR Z9>2048 THEN BEEP
      ● GOTO 600
630 Z9=FNC(Z9)
640 GOTO 500
700 DISP "SET WHICH FLAG (1-2048)";
710 INPUT Z9
720 IF Z9<1 OR Z9>2048 THEN BEEP
      ● GOTO 700
730 Z9=FNS(Z9)
740 GOTO 500
800 DISP "TEST WHICH FLAG (1-2048)";
810 INPUT Z9
820 IF Z9<1 OR Z9>2048 THEN BEEP
      ● GOTO 800
830 IF FNF(Z9) THEN DISP "FLAG SET"
    ELSE DISP "FLAG CLEAR"
840 GOTO 500
850 END
```

Note: The backslash (\) denotes the DIV function which gives the integer portion of the division of the first number by the second.

Lines 10 to 100 initialise the variables and clear all the flags. The F$ string is set to 256 characters giving 2,048 flags. It could just as easily be set to 8 characters (giving 64 flags) or 1,024 characters (giving 8,192 flags). Any number of flags can be chosen by setting the length of the string to the appropriate number of characters.

The flags need not necessarily be cleared at the start of the program. The status of all of the flags can be written to or read from mass storage at any time merely by reading or storing the F$ string.

The function FNS ( ) in lines 120 to 170 is the set flag function. The flag number (Z9) is unaffected by the routine and that flag is set by using the BINIOR function of the I/O ROM.

The function FNC ( ) in lines 180 to 230 is the clear flag function. The flag number is unaffected by the routine and that flag is cleared by using the BINAND function of the I/O ROM.

The function FNF ( ) in lines 240 to 280 uses the BIT function of the I/O ROM to return either a '1' (if set) or '0' (if clear) for the flag number. The flag number is unaffected by the routine.

The program listed in lines 500 to 580 is a short program to demonstrate the capabilities of the three flag functions.

These flag procedures may, hopefully, be useful in programs which require a large amount of conditional testing and branching. They can also be used in simple bit mapping techniques as will be explained next time.

Ken Cronin
C/- Survey Division, M.M.B.W.
625 Lt. Collins Street, Melbourne.

PROGRAMMING TIPS

Desktop Programming Snippets

By Chris Simpson
(03 859 6643)

1. If an HP87 BASIC program requires a BINARY PROGRAM to be loaded, it is normally done so early in the BASIC program with a LOADBIN command. This assumes that we started with a clear (SCRATCHed) memory.
   If however, the program is STOPped and re-RUN (especially likely during program development), an error will occur saying that an attempt to re-load the binary program is illegal. The following trick can obviate the problem:

```
10 ! Fred's program — Rev. 1
20 ON ERROR GO TO start @
   LOADBIN "BIN. PROG"
30 Start: OFF ERROR.
:
```

2. A$ [1,80]= "" or "" → A$ [1,80] is a handy way to fill A$ with eighty spaces, on any BASIC or HPL HP-desktop computer, provided that A$ has been dimensioned to at least 80 bytes in length ..... that is, all EXCEPT HP85 and HP87, for which an anomaly exists! But there is an easy solution:
   10 DIM A$ [80] @ A$ [1,80]= ""
   gives a **null** string for A$
   10 DIM A$ [81] @ A$ [1,80]= ""
   gives 80 spaces in A$
   i.e. dimension to ONE MORE byte than theoretically needed.

HP 85 PROGRAMMING SNIPPET

By Ron Davis
(03 751 1526)

It is very convenient to store numbers in a string when constructing a data base etc., as storage space can be reduced. However there are traps for the unwary, as the following example shows:-

```
A$="********************"
X=-.0333333333
Y=-.0333333333333
A$[3,6]=VAL$(X)
A$[10,13]=VAL$(Y)
A$
**-.03***-3.3*****
Y
-3.33333333333E-2
VAL$(Y)
-3.33333333333E-2
```

```
100 DEF FNS$(T)
200 S8=T ● IF S8=0 OR S8>.001 THEN
    S$[1,5]=VAL$(S8) ● GOTO 600 !S8=short!!!

300 S1=0
400 S8=10*S8 ● S1=S1+1 ● IF S8<10 THEN 400
500 S$[1,2]=VAL$(S8)  ● S$[3,4]="E-"
    ● S$[5,5]=VAL$(S1)
600 FNS$=S$[1,5]
700 FN END
```

ANSI requires that if a number cannot be represented in its full precision in fixed point, then floating point must be used. This is what has happened to Y. Above is a short program segment I concocted to get round the problem by constructing a pseudo-floating point number. (Only positive values were required in my example). S8 was diminsioned as a SHORT variable.

Symbolic Debug/1000

HEWLETT PACKARD

HP Program Development Software product number 92860A

Symbolic Debug/1000 is an interactive, symbolic debugger for source-level FORTRAN and Macro programs on RTE-6/VM and RTE-A.1 based HP 1000 systems. Variables are displayed or modified using names from the original program and the need for load maps and program listings is eliminated. Data types supported are one and two word integer, two, three and four word reals, logical, complex, character, and Hollerith. Symbolic Debug resides in a separate partition from the program being debugged to eliminate program code space intrusion. A single-stepping, source-line capability displays the current and adjacent lines during execution. Conditional breakpoints can be used to monitor variable values and stop the program at a specified value. Using the profiling capability, the user can determine which subroutine is using the most program time and optimize the code to decrease execution time. A small, simple command set, the use of dozens of English error messages, and an on-line "help" facility make Symbolic Debug/1000 a friendly and powerful programmer's productivity tool.

```
   40           RE   = DCOS  (ANG)
   41           IM   = DSIN  (ANG)
   42    2      IF  ( .NOT.  NEW  .AND.  K*KO  .GE.  1 ) GO TO 4
   43 C        ------------------------------------------------
   44 C        COMPUTE TWIDDLES IF NECESSARY ...
   45 C        ------------------------------------------------
   46           U(1) = DCMPLX( RE , -SIGN(IM,DBLE(K)) )
>  47           DO  3  1  =  2,L2N
   48    3      U(I)  =  U(I-1)*U(I-1)
   49           KO = K
   50 C        ------------------
   51 C        BUTTERFLIES.
   52 C        ------------------
   53    4      SHY2 = N
   54           DO 7 STAGE = 1,L2N

DEB.A> b 47/fft
Breakpoint set at 47/FFT
DEB.A> p
Break at 47/FFT
DEB.A> d L2N new re u(1)
L2N = 5       NEW = true      RE = 0.980785282244344
U(1) = (0.980785282244344,-0.195090312760225)
DEB.A> m L2N 6
L2N: 5 => 6
DEB.A>
```


Features

- **Interprets all code types and symbols used**
- **Can display source code during execution (FORTRAN or Macro)**
- **Non-intrusive — no Symbolic Debug code resides in user space**
- **Supports EMA and RTE segmentation**
- **Supports all FORTRAN data types**
- **Program profiler isolates slow subroutines**
- **Source line-by-line single stepping capability**
- **Up to 50 conditional breakpoints to stop program at specified variable value**

Functional description

Symbolic debug. Symbolic Debug recognizes the names, types, and locations of all of the variables and subroutines used in the program, eliminating the need for load maps, symbol table dumps, and mixed listings. The value of a variable can be examined as fast as its name can be typed.

Interactive debugging process. The user interacts with the program as it runs. Variable values may be examined or altered while the program runs without having to insert statements into the code. Bugs can be found faster since there's no need to recompile and load every time a new bug occurs.

Separate partition. Symbolic Debug resides in a 32-page memory partition separate from the program. No code space is lost and no extra statements are added in order to debug. The program being debugged runs exactly the same as it would when not being debugged. No bugs are introduced by the debugger, and more importantly, no bugs disappear when the debugger is present, only to reappear when the debugger is not used. There is no need to restructure a program just to debug it.

Source-level symbolic. Symbolic Debug recognizes what line of source code is about to be executed, and identifies it on the CRT display. Programs can be debugged in the language in which they were written, without the need for inverse assemblies or mixed listings. There is no need to list files at all.

Detects RTE program violations. After a detected violation such as an attempt to access protected memory, memory locations can be examined to determine the cause of the problem. Symbolic Debug pinpoints the line of source code that caused the error, giving the operator an interactive tool for catching system violations.

Standard and conditional breakpoints. Up to 50 breakpoints enable Symbolic Debug to monitor program variables and halt program execution if a variable reaches a specified value. A large number of possible paths can be trapped and values can be quickly tracked through the program to determine where they go wrong.

Supports transfer files and message logging. Non-interactive debugging sessions may occur where users can submit debug commands in a file, and have results logged in another file. This automates the debugging process, so users don't have to wait for bugs whose symptoms may take hours or even days to occur.

Built-in profile monitor. Helps isolate slow parts of the program. High-level analysis of activity distribution within the program helps to identify time-consuming subroutines that should be optimized in order to improve execution time. For example:

Profile for program TEST

Routine	Amount	Histogram
OTEST	39%	**
SUBR	27%	**************************
OTES1	16%	****************
UTILITY	9%	*********
OTES0	3%	***
Other (your code)	2%	**
Other (libraries)	3%	***

Profile for module OTEST:

39% of total time spent here

Line No.	Amount	Histogram
7	20%	********************
8	11%	**********
9	33%	**********************************
13	36%	*************************************

Debug command summary

B <Location>	Sets a breakpoint at location specified.
C<Location>	Clears the breakpoint set at location specified.
D <Locations>	Displays variable.
E	Aborts your program and exits Debug.
G <Location>	Allows your program to proceed from location specified.
I <f1 [f2]>	Executes a set of commands from a file (f1) and optionally logs the output to f2.
L <Location>	Lists a screenful of source code in your program.
M <Loc> <value>	Modifies the value of variable.
P <line>	Allows your program to proceed to the next breakpoint or specified line.
S	Steps to the next line of source code.
T <Location>	Shows location executed without stopping program.
V <number>	Changes the number of source lines displayed on screen.
W	Shows the callers of the current subroutine.

Ordering information

92860A Symbolic Debug/1000

92860A Symbolic Debug/1000, which must be ordered with Use Option 600 or 700, includes:

1. Symbolic Debug/1000 software on Media Option 020, 022, 041, 042, 050 or 051, one of which must be ordered.
2. Symbolic Debug/1000 User's Reference Manual (92860-90001)
3. Symbolic Debug/1000 Configuration Guide (92860-90002)

92860A Media Options

020: Provides 92860A software on 264x Minicartridges

022: Provides 92860A software on 7908/11/12 compatible cartridge tape

041: Provides 92860A software on 1.2M byte flexible disc

042: Provides 92860A software on 5-1/4-inch Minifloppy Disc for Model 6 Microsystem

050: Provides 92860A software on 800 bpi mag tape

051: Provides 92860A software on 1600 bpi mag tape

92860A/92860R Use Options

600: For use in A600 computers

700: For use in A700 or E/F-Series computers

92860R Right to Copy Symbolic Debug/1000 for Use on an Additional Computer System

The 92860R Right to Copy product, which must be ordered with Use Option 600 or 700, is available only to customers who have previously purchased a 92860A product. 92860R consists of:

1. The license to make one copy of software purchased with 92860A for use on an additional computer
2. Symbolic Debug/1000 User's Reference Manual (92860-90001)
3. Symbolic Debug/1000 Configuration Guide (92860-90002)

Support services for Symbolic Debug/1000

92860Q Manual Update Service for 92860A/R

92860FT Customer Support Service (CSS) for 92860A, with Use Option 600 (same media options for updates as for 92860A software)

92860GT Customer Support Service (CSS) for 92860A, with Use Option 700 (same media options for updates as for 92860A software)

92860FV Extension of 92860FT CSS for Symbolic Debug/1000 used on one additional system licensed under 92860R

92860GV Extension of 92860GT CSS for Symbolic Debug/1000 used on one additional system licensed under 92860R

92860FS Software Subscription Service (SSS) for 92860A with Use Option 600 (same media options for updates as for 92860A software)

92860GS Software Subscription Service (SSS) for 92860A with Use Option 700 (same media options for updates as for 92860A software)

92860FW Extension of 92860FS SSS for Symbolic Debug/1000 used on one additional system licensed under 92860R

92860GW Extension of 92860GS SSS for Symbolic Debug/1000 used on one additional system licensed under 92860R

HEWLETT PACKARD

For more information call the HP Sales Office listed in the White Pages. Or write or phone Hewlett-Packard, Data Systems Division, 11000 Wolfe Rd., Cupertino, CA 95014, (408) 257-7000.

Printed in U.S.A. 7/82
5953-2845

HP 1000 USERS' GROUP (N.S.W.) NEWS

DATES OF NEXT MEETINGS

— Meetings will be held regularly on the first Wednesday of each even numbered month i.e. 4 August, 1982, 6 October, 1982, 1 December, 1982 (ANNUAL GENERAL MEETING).
— TIME 4.00 p.m. to 6.00 p.m. with refreshments to follow.
— VENUE is Hewlett-Packard offices at North Ryde.

SITE VISIT

Subject to approval by a Dept. of Public Works/Maritime Services Board SAFETY COMMITTEE we have been invited to visit the site of the PORT KEMBLA COAL LOADER by DASYS P/L.

DASYS have installed a couple of HP1000's to control all aspects of the operation of the coal loader.

Of particular interest to members will be the real time graphics displays and the fiber-optic communication links between equipment.

A Notice will be mailed to each member giving further details of date and transportation arrangements. It is expected that the visit will take the better part of a day including travel to and fro.

OFFICE BEARERS

Corrado DiQual STC P/L	(02) 699 0044	President
Bill Wallace Office of the Supervising Scientist	(02) 387 0681	Secretary
Ngo Hai Lever and Kitchen P/L	(02) 82 0244	Treasurer
Bill Filson DASYS P/L	(02) 43 5150	Committee
Ian Franzman HP P/L	(02) 887 1611	Committee
Shirley Alexander ACA	(02) 699 7181	Committee

1981 IUG Conference Panel questions and answer session

By George Low, DSD

The following items summarize the questions raised at the open panel session at the 1981 IUG Conference at the University of Maryland at the HP responses to these questions.

Question: With CS-80 products, HP now has essentially three lines of disc drives. What is going to happen to the MAC and H types of disc drives?

Response: The MAC (Multi-Access Controller) family of disc drives will continue to be on the HP Price List for 12 to 28 months (7920 about 12-18 months, 7925 about 24-28 months). HP will notify its sales force and customers one year in advance of removing any MAC discs from the price list. The H type discs (7906H, 7920H, 7925H) are planned to be removed from the HP Price List on approximately September 8, 1982.

Question: Super intelligence on peripherals — is this concept going to be in all HP peripheral products (besides the CS-80 discs)?

Response: Yes, the advent of microprocessor intelligence will enable more peripherals to have more diagnostic capabilitiy and much smarter interface capabilities.

Question: Is this intelligence being applied to tape drives? Any new tape drives in the works?

Response: Yes, there are new tape drive projects upcoming, about a year away from introduction.

Question: What about support of the original GRAPHICS 1000 on RTE-6/VM? Many of us have heavy investments in application programs.

Response: The original GRAPHICS 1000 (92840A) is a mature HP software product, and as such, no enhancements are planned to support it on RTE-6/VM. The original GRAPHICS 1000, along with other non-supported software, (e.g., -6940, 2313 interface) may work with RTE-6/VM (we have run BRUNO and PENNEY with the old graphics on RTE-6/VM), we just did not have time to check it out nor planned to implement it. As to your application programs, try them out on RTE-6/VM — you may be pleasantly surprised. If there are any problems, our SEO Organization is ready to help.

Question: The obsolescence of products, e.g., -9885M Flexible Disc, has caused problems because not everyone can afford to keep buying new products that are not backward compatible.

Response: The HP policy is to produce products that maximize benefits to our customers, and we try our best to make new products backwards compatible, within the limit of available resources versus return on investment.

Question: Software updates distribution is totally inadequate.

Response: We are aware of problems in this area and are continuously working to improve distribution channels.

Question: Maintenance bug reports in the SSB are not coming back from the factory. Going through the SE to the factory is a problem — i.e., the SE is too busy to address customer bug problems (teaching class, demos, taking training, etc.).

Response: The HP SE should be your main interface to the factory. If your SE is busy, there normally is a backup to cover accounts. Once the SE contacts the factory, DSD Technical Marketing, and the Lab, have one week to respond to bug reports.

Question: Is MRJE scheduled for the HP 1000?

Response: Yes, MRJE is coming for the HP 1000 in this summer's timeframe.

Question: Will there be hardware floating-point on the XL?

Response: No, but by the time you read this, HP will have announced two new exciting processors, HP 1000 A700 Series (Phoenix) with hardware floating point, and A600 Series (Lightning) with firmware floating point that give you 100 times and 25 times, respectively, the present performance of the XL. Phoenix and Lightning will have a new RTE-A.1 Operating System which will be compatible with present XL programs.

Question: Data Comm on XL via DS is expensive.

Response: Yes, but HP has one of the most powerful distributed systems-networking capability in the industry. We think that we are the leaders in DSN, with over 5000 HP computer systems installed in DS networks throughout the world.

Question: The 7221 Plotter is not supported on GRAPHICS/1000-II?

Response: The 7221, as well as the 7220 and 7225 Plotters, will be supported on GRAPHICS/1000-II on the new 2213 PCO in the April-May, 1982, timeframe.

Question: Do we have to purchase Right-To-Copy on XL each time?

Response: Yes, we offer a standard "R" product (Right-To-Copy to one system) for nearly all software products.

Note: This article was published in the April/May '82 edition of 'INTERFACE 1000'.

Focus 1000

The "VM" in RTE-6/VM explained

By Bill Jacobs
(TUSC Computer Systems)

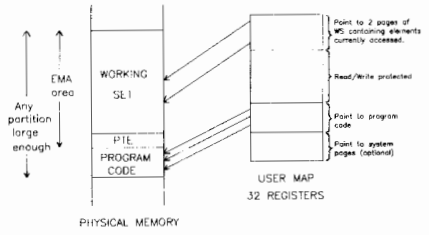
RTE-6/VM brings with it a wealth of innovations and enhancements to RTE-4B. Three of these ("the big three") stand out: Extended code Space for large programs, Shareable EMA and Virtual Memory for data. This article discusses the implementation of the last of these.

The first obvious question is: "How can my program access virtual memory?" Well, it is simplicity itself: just declare your variables to be in EMA and inform the loader that the program needs virtual memory. For example,

```
FTN4X, L
$EMA BIG
     PROGRAM VMX1
     COMMON/BIG/J(1024, 1024), K (1024, 1024), I (4048)
     J (1,1) = 1
     I (4000) = 2
     DO 10 N = N
     K (1,N) = N+1
10   CONTINUE
     :
RU,  LOADR
     /LOADR: OP,VM
     /LOADR: RE, %VMX1
     /LOADR: END
```

will do the trick nicely. (For PASCAL, a $HEAP 2 directive will also be required).

The similarity between EMA and VMA does not end there. For VMA, logical and physical memory look like this:



The working set can be any size, limited, of course, by physical memory. If the size of the EMA variables is less than the working set, then Virtual Memory doesn't enter the game.

The only real change in this scheme from RTE-4B EMA is the one-page PTE. PTE stands for Page Table (don't ask me what the E stands for), and provides information on what data is in the working set. The PTE provides one word of information on each of the (potential) 1024 pages in the working set. Before detailing the structure of the PTE, let's define a new term.

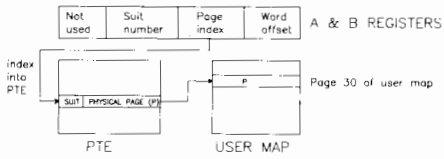
One can define up to 128 Mbytes (or 64 Mwords) of data in virtual memory. Now, 1 Mword is 1024 pages, so we divide virtual memory into 64 groups of 1024 pages. Each 1024-page group is called a "suit" (the reason for this name will become apparent later), and suits are numbered from 0 to 63.

Back at the PTE scene, each word of the PTE relates to one page of the working set, and hence one page of virtual memory. Now comes the key rule: the nth word of the PTE always relates to the nth page of some suit. Hence, if a program accesses a variable on page 50 of EMA (i.e. page 50 of suit 0), this will relate to word 50 of the PTE. An access to page 1044 (page 20 of suit 1) will relate to word 20 of the PTE, and so on.

Keeping this in mind, let's examine the contents of a word of the PTE:

Each word supplies the suitnumber currently being accessed and the physical address in memory of that page. Initially this will be blank. As elements are accessed, these entries are filled in.

So now we can see in total the sequence required to access a virtual memory element:



When a VM element is accessed, the VMA microcode creates a 2 word address in the A and B registers consisting of a suit number (0 to 63), a page number within that suit (0 to 1023) and a word offset within that page (0 to 1023). The page number acts as an index into the PTE. The suit number in the PTE is compared to the suit number of the accessed variable. If they don't agree, a page fault may be required — hang in there, we'll come to that later. If they agree, the physical page address P is placed in map register 30 (remember the last two map registers act as a window into the working set), and this is combined with the word offset to create the final logical address. Whew!

Incidentally, all the routines to do this are microcoded to reduce overhead. In fact, the only VM routines that are software implemented are those dealing with a disc swap (when the software overhead pales into insignificance compared to the disc access time).

As an example, let's go through our sample program. The first access, J(1,1), is on page 0 and suit 0. Hence, word 0 of the PTE is filled with suit 0 and the first available physical page of the working set. The next access, I(4000), is on page 3 of suit 2. Hence, word 3 of the PTE is filled with suit 2 and another physical page of the working set.

Suppose, however, that the working set is full (eg. The 10th page of EMA has been accessed, and there are only 9 pages in the working set). In that case, a random replacement algorithm is used to choose a page to post to disc (and clear its PTE entry) — the page required is brought into memory and noted in the PTE. Random replacement is used because it requires minimum overhead.

In our sample program, trouble looms. The next access is to k(1,1) which is on page 0 of suit 1. Hence PTE word 0 is looked up, but is already assigned (with suit 0). Well, we could post that page to disc to make room, but that seems a waste if there is still room in the working set. Moreover, the program is in a loop alternately accessing page 0 of suit 0 and 1, so a disc swap on every access would be required leading to thrashing. This symptom will occur whenever there is alternate access to elements a multiple of 1024 pages apart. It is called a synonym collision (the Synonym being the page number, within a suit). To overcome this problem, a "synonym table" (STE) is kept in the VMA code with room for 32 entries. When a synonym collision occurs, the offending entry is placed in the STE and a new entry is made in the PTE. Hence, a disc swap is only required if,

(1) The working set is full, or
(2) a synonym collision occurs and the STE is full.

If you're still reading this, I will present an analogy which may clarify the situation. Imagine a pack of cards (that's our virtual memory array — each card represents a page). The pack is divided into four suits: spades, hearts, diamonds and clubs (so that's where the term came from), of thirteen cards each: ace, king ..... two — strictly, there should be 1024 cards per suit.

Next imagine a desk with space on top for 13 cards — one for an ace, one for a king, and so on (that's our PTE). It also has a drawer with room for 32 cards of any suit or value (the STE). Unfortunately, the desk is flimsily constructed, and it can only support the weight of n cards (n is the working set, or available memory). The excess cards are kept in a town a long way away (the disc).

That's the scenario — now for some action. The boss (program) wants to see a card, say the queen of clubs. The secretary (VMA routines) looks for the card on top of the table in the queen position. If it is there, the boss is satisfied.

If, however, another queen is there (say the queen of hearts), the secretary searches through the drawer for the queen of clubs. If found, she swaps the queen of hearts with the queen of clubs and the boss is satisfied. If it is not there, the secretary gets in her car and drives to the town to retrieve the queen of clubs, returns, places the queen of hearts in the drawer (if the drawer had been full, a random card would need to have been removed from the drawer and taken to town), and places the queen of clubs on the desk — the boss is satisfied.

Finally, if there is no queen on the desk, the secretary will need to retrieve it from the town (it could not be in the drawer — why?). If there are already n cards in or on the desk (the working set is full), she chooses a card at random to take and leave in the town. She returns with the queen of clubs, places it on the desk, and the boss is satisfied.

Note that the boss is always satisfied — i.e. VMA works! There are some other possible situations that may arise. I suggest you test the analogy on them.

A couple of final points. The virtual memory on disc is called the "backing store file" Routines are available to manipulate this file — in particular, to open and save it. Hence, a file may already exist which contains the initial values of the arrays. By simply opening this file as the backing store file, a program can fill its VM arrays with data with that single sub-routine call. Similarly the VM array can be saved on disc by just closing the backing store file (by default, it is purged upon program termination).

I mentioned before that, should a page need to be flushed to disc, that page is chosen at random. This is not always true. In the case of sequential access to arrays (i.e. faults caused by a page request in the same direction from the previous fault), it is possible to decrease the disc access time required to flush a page and read in the fault page. This is achieved by choosing a page to flush that can be accessed on the same revolution of the disc as the fault page to be read. The average disc access time is reduced by a factor of 3 in this situation.

HP1000 and Dial-up Modems

If you have tried to set up a dial up connection to a HP1000 you have probably come up against some frustrating problems.

HP's SOLUTION

— Buy a BACCI card (hp12966a) and plug it in your 1000.
— Have an auto answer modem installed by TELECOM DATEL division.
— Connect the card to the modem
— Use modem driver DVAO5.
— Now you can talk to your 1000 from anywhere in the countryside.

HOW IT WORKS

The DVAO5 driver does not allow you access to any of the modem control lines, it does it all for you (smart driver eh!).
When you issue a line open to the card in your WELCOM file the driver sits and waits for RING INDICATOR (RI) from the modem to come true.
When the phone rings, the driver detects RI true and sets DATA TERMINAL READY (DTR) true.
This tells the modem to answer the incoming call.
The driver then waits for RECEIVE LINE SIGNAL DETECT (RLSD) to come true whereupon it enables your terminal and you can log on.
When you finish your session you log off.

NOW THE TROUBLE STARTS

Next time you want to use your 1000 you dial the number and ..... THE PHONE'S BUSY.
Hey somebody else must be using it, try again later.
After repeated efforts to connect to your 1000 and all you keep getting is busy tone, you suspect something is wrong.
So you call up the System manager and he tells you that nobody is using that particular port.
That smart driver (DVAO5) has noticed that RLSD has gone false indicating that the incoming call has been disconnected so it just waits until the connection is re-established but it doesn't tell the modem to hang up the phone by setting DTR false. End result is like leaving the phone off the hook.

ASK FOR HELP

You talk to your friendly HP SE and he mentions that you're not the only one to have had this problem.
He has found a fix by writing a modem monitor program to solve most of your problems and he'll gladly give you a copy.
However, he warns you that there are still some nasties that can't be worked around because that smart driver (DVAO5) won't give you access to the modem control lines.

— If your phone line drops out for any number of reasons the driver will still keep the line open and the phone off the hook.
solutions: re-boot, or turn the modem off then on again.
— If your phone number happens to be the same number as the local pub except that the last two numbers are reversed you'll end up blasting the eardrum of any of the local wives who are calling to find if hubby is still there. What's more, that smart driver is still keeping the phone off the hook waiting for RLSD to come true after that wrong number so you can't call up until the modem is reset.

WORKABLE SOLUTIONS

— The best solution of course is for HP to change the driver to do all the right things, but that would be too much to ask.
— If you happen to be part of a company that designs and makes modems, you have a talk to the modem designer and he draws up a circuit which implements the modem connect/disconnect controls in a black box that you can place in series with the modem.
— Acquire a smart 1200bps full duplex modem which does all the right things.

BONUS POINTS

Now you find that you no longer need to buy BACCI cards for each dial up modem connection, you can also use the box or the smart modem on the 8 channel multiplexer without any supervising programs.
What's more you can now connect non-HP terminals to these mux ports by disabling ENQ/ACK handshaking (DVAO5 would not let you do this).

Corrado DiQual
STANDARD TELEPHONES AND CABLES P/L
(phone: (02) 699 0044).

Auto Answer Modem Monitor Program

By David Triggs, Systems Engineer, Hewlett-Packard, Sydney.

```
ftn4x,l,q
c-----------------------------------------------------------------
c
c This is a monitor program to control the operation of an auto
c answer modem connected to an HP 1000 computer via a baci interface
c card.  The program is hard coded for a particular lu for the baci
c interface, this can easily be changed.
c
c-----------------------------------------------------------------
c
      program modem(),Modem monitor program <820721.1829>
c
c data declarations.
c
      integer dummy, lu, fmgr(3), a, b, eqt5, count
      integer mess1( 17 ), mess2( 11 ), mess3( 11 )
      integer mess4( 11 ), mess5( 12 ), mess6( 10 )
      integer buf( 14 ), unbuf( 14 ), mess( 14 )
      integer hello( 15 ), bye( 28 )
c
c message initialization.
c
      data mess1 /'/MODEM: restarting with line open'/
      data mess2 /'/MODEM: call received'/
      data mess3 /'/MODEM: user logged on'/
      data mess4 /'/MODEM: logon timeout'/
      data mess5 /'/MODEM: user logged off'/
      data mess6 /'/MODEM: line hung up'/
      data hello /'Hewlett Packard 1000 computer'/
      data bye /'Call terminated, please wait 2 seconds before hanging u
     1p'/
c
c paramater specification.
c
      data lu / 56 /
      data fmgr /'FMG56  '/
      data buf /'EQ, 5,BU'/
      data unbuf /'EQ, 5,UN'/
c
c detatch program from session so that it can be run by a user if
c accidently ( or deliberately ) removed.  for example if a user
c wishes to log off and back on again without hanging up or for some
c other reason wishes to remove the 'modem' program it can be 'of'ed and
c run again.
c
      call dtach( dummy )
c
c if a file manager already exists for the modem line then a user
c must be logged on.  also needed to allow removal and re-running of
c 'modem' program.
c
      call idget( fmgr )
      call abreg( a, b )
      if ( a .eq. 0 ) goto 100
      call reio( 2, 1, mess1, 17 )
      goto 350
c
c
c
c
c
c Un-buffer terminal so that line open request will be
c performed with wait.
c
  100 do 125 a = 1,14
          mess( a ) = unbuf( a )
  125 continue
      call messs( mess, 28 )
c
c issue line open.
c
      call exec( 3, 3200b+lu )
c
c enable terminal.
c
      call exec( 3, 2000b+lu )
      call reio( 2, 1, mess2, 11 )
c
c buffer the terminal again.
c
      do 150 a = 1,14
          mess( a ) = buf( a )
  150 continue
      call messs( mess, 28 )
c
c print welcome message.
c
      call reio( 2, lu, dummy, 0 )
      call reio( 2, lu, dummy, 0 )
      call reio( 2, lu, hello, 15 )
c
c wait for user to log on, test for this by watching for an id
c segment to apear for their file manager.  if user does not log on
c within a reasonable time hang up.
c
      count = 0
  200 call idget( fmgr )
      call abreg( a, b )
      if ( a .ne. 0 ) goto 300
      call exec( 12, 0, 2, 0, -10 )
      count = count + 1
      if ( count .lt. 5 ) go to 200
      call reio( 2, 1, mess4, 11 )
      goto 500
  300 call reio( 2, 1, mess3, 11 )
c
c wait for user to log off.  test for this by looking for the id
c segment of their file manager to disapear.
c
  350 call idget( fmgr )
      call abreg( a, b )
      call exec( 12, 0, 2, 0, -10 )
      if ( a .eq. 0 ) goto 350
c
c check that user has really logged off and that they haven't just
c rescheduled their file manager.
c
  400 call idget( fmgr )
      call abreg( a, b )
      if ( a .ne. 0 ) goto 350
      call reio( 2, 1, mess5, 12 )
c
c
c
c print termination message.
c
      call reio( 2, lu, dummy, 0 )
      call reio( 2, lu, bye, 28 )
c
c issue line close.
c
  500 call exec( 3, 3100b+lu, 0 )
      call reio( 2, 1, mess6, 10 )
c
c go back and wait for next call
c
      goto 100
c
c program never termintes, the system can be safely shut down
c without removing it.  it can be removed with 'of,modem' if needed
c
      end
```

Systems Improvement Committee Organizes

The Systems Improvement Committee is now ready to serve members of the HP 1000 International Users Group.

The committee, comprised of several willing volunteers, will review systems improvements that are suggested by HP 1000 IUG members. In turn, these suggestions will be submitted to Hewlett-Packard in hopes of redesigning hardware and software, resolving bugs in the system, and helping to develop new products.

At the last board meeting in Princeton, New Jersey, the Board discussed the organization of the committee. It was decided to encourage local user group participation by urging members to submit systems-improvement suggestions to one designated representative of the IUG. The representative would then compile the suggestions, and forward them on to the Systems Improvement Committee chairperson. The Systems Improvement Committee will review the suggestions for validity and then submit them to HP.

Greg Gillen, manager, Technical Marketing Department, DSD, has suggested that systems-improvement requests be compiled and prioritized. He also recommends a bi-annual submission of the requests, in January/February and August/September, to HP.

An effective way to fulfill both of these requirements may be to publish a list of proposed enhancements in Interface, so that members can vote on which enhancements are most important. Users group opinion could thus be presented en masse, in a single, well-documented input, twice a year.

SYSTEMS IMPROVEMENT COMMITTEE

Chairperson
Glen Mortensen
Senior Engineer
Intermountain Technologies, Inc.
1400 Benton Street
P.O. Box 1604
Idaho Falls, Idaho 83401

Sandy Wolfe
Senior Staff Engineer
Hughes Aircraft
800 Apollo Street
P.O. Box 902
Building E51, Mail Station A264
El Segundo, California 90245

Roger W. Bailey
Systems Programmer
Amdahl Corporation
1250 E. Arques
Sunnyvale, California 94086

Judith Skinner
Software Specialist
McInnis & Associates
3317 N. Shartel Avenue
Oklahoma City, Oklahoma 73118

Ted Varga
Systems Analyst
Pittsburgh-Des Moines Steel Company
P.O. Box 1447
Provo, Utah 84601

Sheryl Coppenger
Tennessee Valley Authority
152 Liberty Building
Knoxville, Tennessee 37902

Duncan McIntosh
Vice-President Engineer
Spectrum Planning, Inc.
1850 North Greenville Avenue
Richardson, Texas 75081

Norman Kay
CSIRO Division of Chemical Physics
P.O. Box 160
Clayton, Victoria 3168
Australia

Glenn A. Simpson
Specialist, Engineering Systems
Westinghouse Canada, Inc.
Box 510, HO-7
Hamilton, Ontario L8N 3K2
Canada

Raymond R. Gingras
Scientific Programmer/Analyst
Factory Mutual Engineering & Research
1151 Boston-Providence Turnpike
Norwood, Massachusetts 02062

David Marr
Western Electric Senior Engineer
Bell Laboratories
Department 46125 Room 4A-330
Whippany, New Jersey 07981

Nate Wolk
Hughes Aircraft
Building 265, MS P50
8433 Fallbrook Avenue
Canoga Park, California 91304

Harvey Packard
Engineer
Air Force Weapons Laboratory
2941 Valencia NE
Albuquerque, New Mexico 87110

Thomas G. Pitts
Senior Research Engineer
Babcock & Wilcox — LRC
P.O. Box 1260
Lynchburg, Virginia 24505

John J. O. Evans
Ken Harris
Smiths Industries
Winchester Road
Basingstohe, Hampshire
England

Note: This article was published in the April/May '82 edition of 'Interface 1000'. Any comments or enquiries should be directed to Norm Kay, CSIRO, (03) 544 0633.

COMING EVENTS

Sept. 6	RTE-6/VM Upgrade course, HP Melbourne.
Sept. 6	PASCAL/1000 course, HP Sydney.
Sept. 8	H.P.D.C.U.G.V. (Desktop Users' Group) Meeting — Mass storage techniques.
Sept. 13	IMAGE/DBMS course, HP Melbourne.
Sept. 20	HP 85 Basic Programming course, HP Melbourne.
Sept. 20	RTE-6/VM Session Monitor course, HP Sydney.
Sept. 27	MACRO/1000 course, HP Melbourne.
Sept. 28	Canberra Technical Users' Group meeting, 8 p.m. Data Science.
Sept. 30	Language Workshop — PASCAL, 9 am. - 5 pm. HP Melbourne.
Oct. 4	RTE-6/VM Session Monitor course, HP Melbourne.
Oct. 4	HP Basic Programming, HP Melbourne.
Oct. 6	HP1000 Users' Group (N.S.W.) meeting, 4 - 6 p.m. at HP Sydney.
Oct. 11	HP 85 Basic Programming, HP Sydney.
Oct. 18	RTE-6/VM System Manager course, HP Melbourne.
Oct. 18	RTE-6/VM System Manager course, HP Sydney.
Oct. 25	HP 9826/36 Operating & Programming course, HP Melbourne.
Oct. 25	IMAGE/DBMS course, HP Sydney.
Nov. 4	H.P.D.C.U.G.V. meeting — Data acquisition using Desktop Computers.

CLASSIFIED ADVERTISEMENTS

FOR SALE

9845A Desktop Computer with '64' K memory & Mass Memory ROM
9885M Floppy disk drive (8")
9825A Desktop Computer with '24k' memory & lots of ROMS
9831A Desktop Computer with Mass Memory ROM
9869A Card Reader

Enquiries should be directed to:
Chris Simpson
Phone: (03) 859 6643

HP DEMO STOCK
9874A Digitizer. $12,904
98439A Assembly Execution & Development ROM. $2,025
98046B RS232C Data Comm. $1,051
7225/004/17601A Plotter. $3,038
98032A 16 Bit/Interface. $525

All enquiries should be directed to:
Greg Sadler
Hewlett-Packard Aust. Ltd.
31-41 Joseph Street,
Blackburn. Vic. 3130.
Phone: (03) 877 7777

H.P.D.C.U.G.V. MEMBERS

Following the adoption of the Constitution, your annual subscription fee of $10 is now due and payable.

Send your cheques to -
Ron Davis,
Rossmore House,
Ridge Road,
MT. DANDENONG, 3767

THIS IS YOUR GROUP!