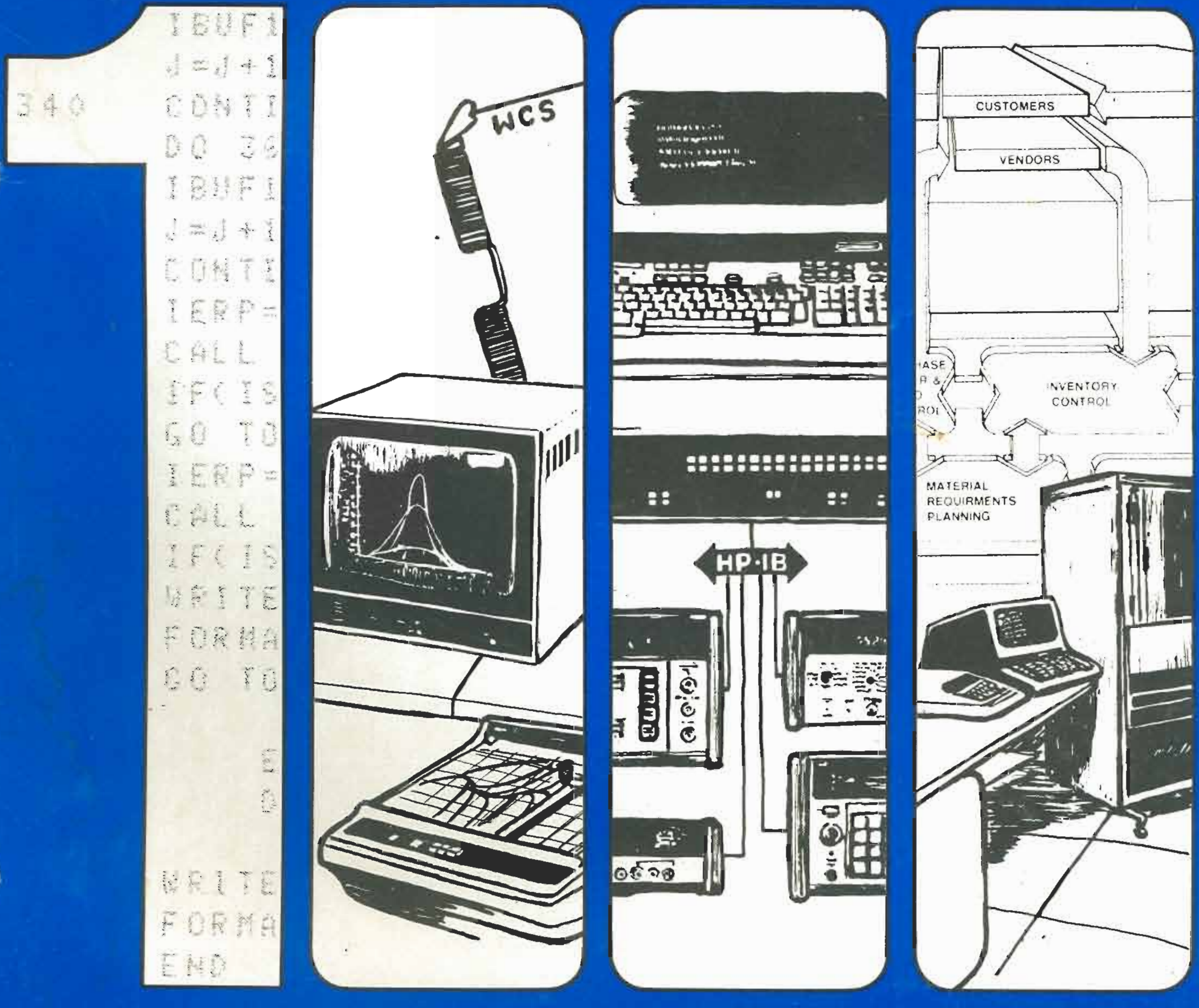
Hewlett-Packard Computer Systems COMMUNICATOR



1979

Volume III Issue 3

HEWLETT-PACKARD
COMPUTER SYSTEMS

Volume III
Issue 3

COMMUNICATOR/1000

Feature Articles

Departments

EDITOR'S DESK

ABOUT THIS ISSUE

Issue 3 of the Communicator has articles of interest to a broad spectrum of readers. Included are two feature articles in the area of Instrumentation, an article in the Operations Management category, and an Operating Systems article.

For those readers interested in using the HP-IB to automate some aspect of a production environment, Larry Marafka of Westinghouse has contributed an article which tells the story of how his company used the HP-IB to automate a rotor balancing operation. Larry's article covers the entire scope of the project from the original cost and benefit considerations, through the implementation, to the final evaluation of the results, and considerations for expanding the system. Also of interest to HP-IB users will be Pete Almeroth's article, "Using SRQ to Schedule Programs". Pete's a Systems Engineer in HP's Cleveland office. His article discusses the considerations that must be made when using an HP-IB service request to schedule execution of a program. Pete's article includes a flowchart detailing all of the options that are involved in SRQ scheduling of programs. For those readers interested in learning more about the HP-IB, this issue contains plenty of good reading.

In the Operations Management category Frank Fulton, an HP customer working for R. J. Reynolds Tobacco Co. in North Carolina has submitted a fine article on Data Acquisition. Frank's article describes an efficient and easily maintainable method for collecting data from persons not trained in the use of computers. His article provides a very friendly operator interface, and for protection of the system. The methods are not only practical, but versatile as well, and will make interesting reading.

The Operating System category contains an excellent article by Glenn Talbott from HP's Data Systems Division. Glenn's article "Recovering System Resources via Automatic Timed Reboot" shows an easy way to keep system resources as available as possible, thereby keeping throughput at its maximum. Any reader that has a system used for a variety of purposes, by a variety of users, who sometimes forget to release ID segments (or some other resource), could surely benefit by Glenn's solution to this common problem.

Again in this issue, the choice of the winners of the HP-32E calculators was not an easy one. After reviewing the eligible articles, three judges from the Technical Marketing Department at DSD choose the following as calculator winners, based on technical content, general interest, completeness and clarity:

For a feature article from a customer	**SCHEDULING DATA ACQUISITION PROGRAMS WITHOUT PROVIDING GENERAL SYSTEM ACCESS TO TERMINAL USERS** Frank Fulton
For a feature article from HP field personnel	**HP-IB AND THE SRQ** Pete Almeroth

Congratulations to the Winners!

It was a pleasure to be able to put two articles contributed by customers into this issue. Thanks to all who contributed.

Sincerely,

The Editor

BECOME A PUBLISHED AUTHOR IN THE COMMUNICATOR/1000 . . .

The COMMUNICATOR is a technical publication designed for HP 1000 computer users. Through technical articles, the direct answering of customers' technical questions, cataloging of contributed user programs, and publication of new product announcements and product training schedules, the COMMUNICATOR strives to help each reader utililize their HP 1000's more effectively.

The Feature Articles are clearly the most important part of the COMMUNICATOR. Feature Articles are intended to promote a significant cross-fertilization of ideas, to provide in-depth technical descriptions of application programs that could be useful to a wide range of users, and to increase user understanding of the most sophisticated capabilities designed into HP software. You might think of the COMMUNICATOR as a publication which can extend your awareness of HP 1000's to include that of thousands of users worldwide as well as that of many HP engineers in Data Systems factories at Cupertino, California and Grenoble, France.

To accomplish these goals, editors of the COMMUNICATOR actively seek technical articles from HP 1000 customers, HP Systems Engineers in the Field, and Marketing and R&D Engineers in the factories. Technical articles from customers are most highly valued because it is customers who are closest to real-world applications.

WIN AN HP-32E CALCULATOR!

Authoring a published article provides a uniquely satisfying and visible feeling of accomplishment. To provide a more tangible benefit, however, HP gives away three free HP-32E hand-held calculators to Feature Article authors in each COMMUNICATOR/1000 issue! Authors are divided into three categories. A calculator is awarded to the author of the best Feature Article in each of the author categories. The three author categories are:

1. HP 1000 Customers;

2. HP employees not in Data Systems Division (e.g., HP Systems Engineers, users in other HP Divisions, etc.);

3. HP Data Systems Division employees not in the Technical Marketing Dept. (from which the COMMUNICATOR Editor is chosen).

Each author category is judged separately. A calculator prize will be awarded even if there is only one entry in an author category.

Feature Articles are judged on the following bases: (1) quality of technical content; (2) level of interest to a wide spectrum of COMMUNICATOR/1000 readers; (3) thoroughness with which subject is covered; and, (4) clarity of presentation.

What is a Feature Article? A Feature Article meets the following criteria:

1. Its topic is of general technical interest to COMMUNICATOR/1000 readers;

2. The topic falls into one of the following categories —

   OPERATING SYSTEMS
   DATA COMMUNICATIONS
   INSTRUMENTATION
   COMPUTATION
   OPERATIONS MANAGEMENT

3. The article covers at least two pages of the COMMUNICATOR/1000, exclusive of listings and illustrations (i.e., at least 1650 words).

There is a little fine print with regard to eligibility for receiving a calculator; it follows. No individual author will be awarded more than one calculator in a calendar year. In the case of multiple authors, the calculator will be awarded to the first listed author of the winning article. An article which is part of a series will compete on its own merits with other articles in the issue. The total of all articles in the series will not compete against the total of all articles in another series. Employees of Technical Marketing at HP's Data Systems Division factory in Cupertino are not eligible to win a calculator.

All winners of calculators will be announced in the issue of the COMMUNICATOR/1000 in which their articles appear. Again, all Feature Articles are judged by an impartial panel of three DSD Technical Marketing Engineers.

A SPECIAL DEAL IN THE OEM CORNER

When an HP 1000 OEM writes a Feature Article that is not only technically detailed and insightful but also application-oriented as opposed to theoretical, then that OEM may ask that the article be included in THE OEM CORNER. A Feature Article included in THE OEM CORNER may contain up to 150 words of pure product description as well as a picture or illustration of the OEM'S product or its unique contribution. HP's objective is twofold: (1) to promote awareness of the capabilities HP 1000 OEMs' products among all HP 1000 users; and, (2) to publish an article of technical interest and depth.

IF YOU'RE PRESSED FOR TIME . . .

If you are short of time, but still have that urge to express yourself technically, don't forget the COMMUNICATOR/1000 BIT BUCKET. It's the perfect place for a short description of a routine you've written or an insight you've had.

THE MECHANICS OF SUBMITTING AN ARTICLE

If at all possible please submit an RTE File containing the text of your article recorded on a Minicartridge (preferrably) or on a paper tape along with the line printer or typed copy of your article. This will help all of us to be more efficient. The Minicartridge will be returned to you promptly. Please include your address and phone number along with your article.

All articles are subject to editorship and minor revisions. The author will be contacted if there is any question of changing the information content. Articles requiring a major revision will be returned to the author with an explanatory note and suggestions for change. We hope not to return any articles at all; if we do, we would like to work closely with the author to improve the article. HP does, however, reserve the right to reject articles that are not technical or that are not of general interest to COMMUNICATOR/1000 readers.

Please submit your COMMUNICATOR/1000 article to the following address:

Editor, COMMUNICATOR/1000
Data Systems Division
Hewlett-Packard Company
11000 Wolfe Road
Cupertino, California 95014
USA

The Editor looks forward to an exciting year of articles in the COMMUNICATOR/1000.

With best regards,

The Editor

LOCUS ANNOUNCEMENTS

The following are changes to existing programs listed in the Data Systems LOCUS Program Catalog (22000-90099).

The "WOLF" program has been revised. It is now available on magnetic tape:

22682-10993	800 bpi magnetic tape	$95.00
22682-11993	1600 bpi magnetic tape	$95.00

NOTE

The article entitled "LOCUS CHANGES and ADDITIONS" that appeared in Volume III, Issue 2 of the Communicator has an error in the price of the "Pascal-S Compiler/Interpreter" program. The correct price is $100.00 as previously defined in Volume II, Issue 5 of the Communicator.

LETTERS TO THE EDITOR

Dear Editor:

In the past few months, I have found two tidbits in the File Manager that I have always wanted to see, but never knew about. In fact I cannot find them in the documentation anywhere. They are listed below:

- A "wild card" letter exists in file names for a directory list.
- A line number range is specified for file lists.

I am wondering how many other useful features there are in the File Manager that are not documented. These features would make a good Communicator article.

I am also wondering what the circulation of the Communicator is.

Yours truly,

Paul A Swartz
Computer Systems Consultants
Tucson, Arizona

Dear Mr. Swartz,

In your search for these two File Manager "tidbits", you must have overlooked the Batch-Spool Monitor Reference Manual. The "wild card" letter is described on page 2-91 under the heading ":DL,*namr* format". The line number range is described on page 2-31.

Our manual was most recently revised in December of 1978, but earlier versions will also contain this information.

In answer to your second inquiry, we are pleased to be able to tell you that we have over 2600 readers of the Communicator. These readers include customers, HP field personnel, 1000 users in other HP divisions, and DSD factory personnel.

Thank you Mr. Swartz for your input. It is our hope that we have documented all of the features of the File Manager. However, if you do come across any "tidbits" that are not documented, or if you find some new and unusual ways to use what is documented, please let us know, so that other readers may benefit also.

With best regards,

The Editor

Dear Editor:

Regarding Martha Robrahn's fine article on shared EMA (Volume II, Issue 6): a note of caution....

Consider a system with only one partition large enough for SEMA1 to run in, and other partitions of proper size for SEMA2. After the programs have executed once, SEMA2's ID segment word 28 indicates that the program has the same EMA size as SEMA1. This is fine, until you attempt (as my customer did), to execute the programs again. This time SEMA1 executes, locks into memory, and attempts to schedule SEMA2. SEMA2 now requires more memory than before (ID segment word 28) and cannot execute. SEMA2 becomes scheduled status one forever.

The answer, obviously, is to return SEMA2's ID segment word 28 back to its original state. This can be accomplished by calling the EMASZ routine to restore ID segment word 28 back to its original value.

This nifty (though unsupported) RTE feature has proven to be the solution to my customer's problem for very large "System Common". Thanks to Martha for this excellent article.

Sincerely,

Frank Hunt
Systems Engineer
HP Neely Tualatin

Dear Mr. Hunt,

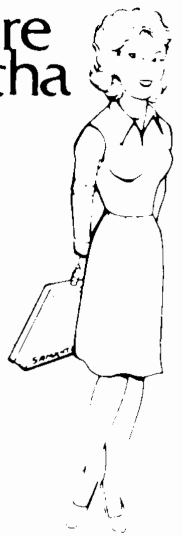
Thanks to you for not only pointing out the problem that your customer had in using shared EMA, but also for pointing out this "nifty" solution. I am sure it will be of some help to other Communicator readers.

Sincerely,

The Editor

BIT BUCKET

Software Samantha



Software Samantha
HP 1000 Communicator
Hewlett-Packard Data Systems Division
11000 Wolfe Road, Cupertino, California 95014

Dear Readers,

Recently I have had numerous inquiries about a complete list of microcode entry points. To make your future generations somewhat easier, below are listed the most current entry points for the M-Series, the E-Series, and the F-Series computers. In the future, a complete, up-to-date list of RP's will be published in the Software Update Notice (SUN). Happy generating!

Sincerely,

Samantha

```
************************************************
*                                              *
*           ENTRY POINT CHANGES                *
*           FOR THE 21-MX SERIES               *
*                                              *
************************************************
*
*     ***** INTEGER ARITHMETIC ENTRY POINTS *****
*
.MPY,RP,100200       * INTEGER MULTIPLY
.DIV,RP,100400       * INTEGER DIVIDE
.DLD,RP,104200       * DOUBLE LOAD
.DST,RP,104400       * DOUBLE STORE
*
*     ***** EAU AND HFP ENTRY POINTS *****
*
.FAD,RP,105000       * FLOATING POINT ADD
.FSB,RP,105020       * FLOATING POINT SUBTRACT
.FMP,RP,105040       * FLOATING POINT MULTIPLY
.FDV,RP,105060       * FLOATING POINT DIVIDE
IFIX,RP,105100       * REAL TO INTEGER FIX
FLOAT,RP,105120      * INTEGER TO REAL FLOAT
*
*    ***** MOVE & COMPARE WORDS ******
*
.MVW,RP,105777       * MOVE WORDS
.CMW,RP,105776       * COMPARE WORDS
*
*    ***** BIT & BYTE INSTRUCTIONS *****
*
.CBT,RP,105766       * COMPARE BYTES
.LBT,RP,105763       * LOAD BYTE
.SBT,RP,105764       * STORE BYTE
.MBT,RP,105765       * MOVE BYTES
.SFB,RP,105767       * SCAN FOR BYTE
.CBS,RP,105774       * CLEAR BITS
.SBS,RP,105773       * SET BITS
.TBS,RP,105775       * TEST BITS
*
*     CLRIO IS GENERATED BY THE COMPILER, BUT IS NOT USED IN RTE.
*     THEREFORE IT'S ENTRY POINT IS MERELY AN RSS (UNCONDITIONAL SKIP).
*
CLRIO,RP,2001
*
*      Z$DBL IS AN ENTRY POINT USED BY THE FTN4 COMPILER (REV 1901 OR
*      LATER)
*      IF IT CONTAINS 3, THEN DOUBLE PRECISION VALUES WILL BE 3 WORDS
*      IF IT CONTAINS 4, THEN DOUBLE PRECISION VALUES WILL BE 4 WORDS
*
Z$DBL,RP,3
*
```

```
*     ***** FFP ENTRY POINTS *****
*
DBLE,RP,105201      * CONVERT REAL TO EXTENDED REAL
SNGL,RP,105202      * CONVERT EXTENDED REAL TO REAL
.DFER,RP,105205     * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.XPAK,RP,105206     * NORMALIZE, ROUND AND PACK WITH EXPONENT AN EXTENDED
*                   * REAL MANTISSA
.XCOM,RP,105215     * COMPLEMENT AN EXTENDED REAL UNPACKED MANTISSA IN
*                   * PLACE
..DCM,RP,105216     * COMPLEMENT AN EXTENDED REAL
DDINT,RP,105217     * TRUNCATE AN EXTENDED REAL
.XFER,RP,105220     * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.GOTO,RP,105221     * TRANSFER CONTROL TO LOCATION
..MAP,RP,105222     * COMPUTE THE ADDRESS OF A 2 OR 3D ARRAY ELEMENT
.ENTR,RP,105223     * TRANSFER THE TRUE ADDRESS OF PARAMETERS USED IN A
*                   * SUBROUTINE CALL
.ENTP,RP,105224     * SAME AS .ENTR, EXCEPT MUST BE THIRD INSTRUCTION
*                   * AFTER THE ENTRY POINT
.PWR2,RP,105225     * CALCULATE REAL X AND INTEGER N, Y=X*2**N
.FLUN,RP,105226     * UNPACK REAL (EXPONENT IN A, LOWER PART OF MANTISSA
*                   * IN B)
*
$SETP,RP,105227     * SET UP A LIST OF POINTERS
*                   * NOTE: $SETP REPLACES .SETP AS OF 1913
*
.PACK,RP,105230     * CONVERT SIGNED MANTISSA OF REAL INTO NORMALIZED REAL
*                   * FORMAT
.XADD,RP,105213     * EXTENTED REAL ADDITION      (IN E AND M SERIES ONLY)
.XSUB,RP,105214     * EXTENDED REAL SUBTRACTION   (IN E AND M SERIES ONLY)
.XMPY,RP,105203     * EXTENDED REAL MULTIPLY      (IN E AND M SERIES ONLY)
.XDIV,RP,105204     * EXTENDED REAL DIVIDE        (IN E AND M SERIES ONLY)
*
**********************************************************************
*
*     XADD, XSUB, XMPY AND XDIV ARE USED FOR FTN INTERFACES
*
XADD,RP,105207      * EXTENDED REAL ADDITION      (IN E AND M SERIES ONLY)
XSUB,RP,105210      * EXTENDED REAL SUBTRACTION   (IN E AND M SERIES ONLY)
XMPY,RP,105211      * EXTENDED REAL MULTIPLCATION (IN E AND M SERIES ONLY)
XDIV,RP,105212      * EXTENDED REAL DIVISION      (IN E AND M SERIES ONLY)
*
**********************************************************************
```

```
**************************************************
*                                                *
*           ENTRY POINT CHANGES                  *
*           FOR THE 21-E SERIES                  *
*                                                *
**************************************************
*
*     ***** INTEGER ARITHMETIC ENTRY POINTS *****
*
.MPY,RP,100200      * INTEGER MULTIPLY
.DIV,RP,100400      * INTEGER DIVIDE
.DLD,RP,104200      * DOUBLE LOAD
.DST,RP,104400      * DOUBLE STORE
*
*     ***** EAU ENTRY POINTS *****
*
.FAD,RP,105000      * FLOATING POINT ADD
.FSB,RP,105020      * FLOATING POINT SUBTRACT
.FMP,RP,105040      * FLOATING POINT MULTIPLY
.FDV,RP,105060      * FLOATING POINT DIVIDE
IFIX,RP,105100      * REAL TO INTEGER FIX
FLOAT,RP,105120     * INTEGER TO REAL FLOAT
*
*    ***** MOVE & COMPARE WORDS ******
*
.MVW,RP,105777      * MOVE WORDS
.CMW,RP,105776      * COMPARE WORDS
*
*    ***** BIT & BYTE INSTRUCTIONS *****
*
.CBT,RP,105766      * COMPARE BYTES
.LBT,RP,105763      * LOAD BYTE
.SBT,RP,105764      * STORE BYTE
.MBT,RP,105765      * MOVE BYTES
.SFB,RP,105767      * SCAN FOR BYTE
.CBS,RP,105774      * CLEAR BITS
.SBS,RP,105773      * SET BITS
.TBS,RP,105775      * TEST BITS
*
*     CLRIO IS GENERATED BY THE COMPILER, BUT IS NOT USED IN RTE.
*     THEREFORE IT'S ENTRY POINT IS MERELY AN RSS (UNCONDITIONAL SKIP).
*
CLRIO,RP,2001
*
*      Z$DBL IS AN ENTRY POINT USED BY THE FTN4 COMPILER (REV 1901 OR
*      LATER)
*      IF IT CONTAINS 3, THEN DOUBLE PRECISION VALUES WILL BE 3 WORDS
*      IF IT CONTAINS 4, THEN DOUBLE PRECISION VALUES WILL BE 4 WORDS
*
Z$DBL,RP,3
*
```

```
*      ***** FFP ENTRY POINTS *****
*
DBLE,RP,105201        * CONVERT REAL TO EXTENDED REAL
SNGL,RP,105202        * CONVERT EXTENDED REAL TO REAL
.DFER,RP,105205       * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.XPAK,RP,105206       * NORMALIZE, ROUND AND PACK WITH EXPONENT AN EXTENDED
*                     * REAL MANTISSA
.XCOM,RP,105215       * COMPLEMENT AN EXTENDED REAL UNPACKED MANTISSA IN
*                     * PLACE
..DCM,RP,105216       * COMPLEMENT AN EXTENDED REAL
DDINT,RP,105217       * TRUNCATE AN EXTENDED REAL
.XFER,RP,105220       * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.GOTO,RP,105221       * TRANSFER CONTROL TO LOCATION
..MAP,RP,105222       * COMPUTE THE ADDRESS OF A 2 OR 3D ARRAY ELEMENT
.ENTR,RP,105223       * TRANSFER THE TRUE ADDRESS OF PARAMETERS USED IN A
*                     * SUBROUTINE CALL
.ENTP,RP,105224       * SAME AS .ENTR, EXCEPT MUST BE THIRD INSTRUCTION
*                     * AFTER THE ENTRY POINT
.PWR2,RP,105225       * CALCULATE REAL X AND INTEGER N, Y=X*2**N
.FLUN,RP,105226       * UNPACK REAL (EXPONENT IN A, LOWER PART OF MANTISSA
*                     * IN B)
*
$SETP,RP,105227       * SET UP A LIST OF POINTERS
*                     * NOTE: $SETP REPLACES .SETP AS OF 1913
*
.PACK,RP,105230       * CONVERT SIGNED MANTISSA OF REAL INTO NORMALIZE REAL
*                     * FORMAT
.CFER,RP,105231       * MOVE 4 WORDS (COMPLEX TRANSFER) (IN F SERIES AND E
*                     * SERIES WITH THE UPDATED FIRMWARE) (SEE END OF LISTING
*                     * FOR PART NUMBER OF THE UPDATED FIRMWARE)
.XADD,RP,105213       * EXTENTED REAL ADDITION      (IN E AND M SERIES ONLY)
.XSUB,RP,105214       * EXTENDED REAL SUBTRACTION   (IN E AND M SERIES ONLY)
.XMPY,RP,105203       * EXTENDED REAL MULTIPLY      (IN E AND M SERIES ONLY)
.XDIV,RP,105204       * EXTENDED REAL DIVIDE        (IN E AND M SERIES ONLY)
*
************************************************************************
*
*      XADD, XSUB, XMPY AND XDIV ARE USED FOR FTN INTERFACES
*
XADD,RP,105207        * EXTENDED REAL ADDITION      (IN E AND M SERIES ONLY)
XSUB,RP,105210        * EXTENDED REAL SUBTRACTION   (IN E AND M SERIES ONLY)
XMPY,RP,105211        * EXTENDED REAL MULTIPLCATION (IN E AND M SERIES ONLY)
XDIV,RP,105212        * EXTENDED REAL DIVISION      (IN E AND M SERIES ONLY)
*
************************************************************************
*
*
*      ***** EMA ENTRY POINTS (F AND E SERIES ONLY) *****
*
.EMAP,RP,105257       * RESOLVE REFERENCES TO EMA ELEMENTS
.EMIO,RP,105240       * USED FOR I/O FROM EMA ARRAYS
MMAP,RP,105241        * MAPS PHYSICAL PAGES INTO LOGICAL ADDRESS SPACE
*
*
************************************************************************
*
* THE PART NUMBERS FOR THE MOST CURRENT FIRMWARE FOR E SERIES COMPUTERS
* ARE AS FOLLOWS:
*
*               13306 - 80013
*               13306 - 80014
*               13306 - 80015
*               13306 - 80016
*               13306 - 80017
*               13306 - 80018
*                5090 - 0589
*                5090 - 0590
*                5090 - 0591
*
************************************************************************
*

*
```

```
***********************************************************************
*
*     THE VIS LIBRARY IS USED IN THE F SERIES ONLY, IT ALLOWS THE
*     ACCESS OF THE TWO WORD VIS MICRO CODE. DO NOT CONFUSE IT WITH
*     SOFTWARE MODULES FOR VIS MICRO CODE (THEY DO NOT EXIST).
*
REL,%VLIB            * VIS SOFTWARE INTERFACE LIBRARY
*
***********************************************************************
*
```

```
************************************************
*                                              *
*           ENTRY POINT CHANGES                *
*           FOR THE 21-F SERIES                *
*                                              *
************************************************
*
*     ***** INTEGER ARITHMETIC ENTRY POINTS *****
*
.MPY,RP,100200       * INTEGER MULTIPLY
.DIV,RP,100400       * INTEGER DIVIDE
.DLD,RP,104200       * DOUBLE LOAD
.DST,RP,104400       * DOUBLE STORE
*
*     ***** EAU AND HFP ENTRY POINTS *****
*
.FAD,RP,105000       * FLOATING POINT ADD
.FSB,RP,105020       * FLOATING POINT SUBTRACT
.FMP,RP,105040       * FLOATING POINT MULTIPLY
.FDV,RP,105060       * FLOATING POINT DIVIDE
IFIX,RP,105100       * REAL TO INTEGER FIX
FLOAT,RP,105120      * INTEGER TO REAL FLOAT
.FIXD,RP,105104      * REAL TO DOUBLE INTEGER FIX    (IN F SERIES ONLY)
.FLTD,RP,105124      * REAL TO DOUBLE INTEGER FLOAT  (IN F SERIES ONLY)
*
*    ***** MOVE & COMPARE WORDS ******
*
.MVW,RP,105777       * MOVE WORDS
.CMW,RP,105776       * COMPARE WORDS
*
*    ***** BIT & BYTE INSTRUCTIONS *****
*
.CBT,RP,105766       * COMPARE BYTES
.LBT,RP,105763       * LOAD BYTE
.SBT,RP,105764       * STORE BYTE
.MBT,RP,105765       * MOVE BYTES
.SFB,RP,105767       * SCAN FOR BYTE
.CBS,RP,105774       * CLEAR BITS
.SBS,RP,105773       * SET BITS
.TBS,RP,105775       * TEST BITS
*
*     CLRIO IS GENERATED BY THE COMPILER, BUT IS NOT USED IN RTE.
*     THEREFORE IT'S ENTRY POINT IS MERELY AN RSS (UNCONDITIONAL SKIP).
*
CLRIO,RP,2001
*
*      Z$DBL IS AN ENTRY POINT USED BY THE FTN4 COMPILER (REV 1901 OR
*      LATER)
*      IF IT CONTAINS 3, THEN DOUBLE PRECISION VALUES WILL BE 3 WORDS
*      IF IT CONTAINS 4, THEN DOUBLE PRECISION VALUES WILL BE 4 WORDS
*
Z$DBL,RP,3
*
```

```
*     ***** FFP ENTRY POINTS *****
*
DBLE,RP,105201        * CONVERT REAL TO EXTENDED REAL
SNGL,RP,105202        * CONVERT EXTENDED REAL TO REAL
.DFER,RP,105205       * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.XPAK,RP,105206       * NORMALIZE, ROUND AND PACK WITH EXPONENT AN EXTENDED
*                     * REAL MANTISSA
.XCOM,RP,105215       * COMPLEMENT AN EXTENDED REAL UNPACKED MANTISSA IN
*                     * PLACE
..DCM,RP,105216       * COMPLEMENT AN EXTENDED REAL
DDINT,RP,105217       * TRUNCATE AN EXTENDED REAL
.XFER,RP,105220       * 3 WORD MOVE (EXTENDED REAL TRANSFER)
.GOTO,RP,105221       * TRANSFER CONTROL TO LOCATION
..MAP,RP,105222       * COMPUTE THE ADDRESS OF A 2 OR 3D ARRAY ELEMENT
.ENTR,RP,105223       * TRANSFER THE TRUE ADDRESS OF PARAMETERS USED IN A
*                     * SUBROUTINE CALL
.ENTP,RP,105224       * SAME AS .ENTR, EXCEPT MUST BE THIRD INSTRUCTION
*                     * AFTER THE ENTRY POINT
.PWR2,RP,105225       * CALCULATE REAL X AND INTEGER N, Y=X*2**N
.FLUN,RP,105226       * UNPACK REAL (EXPONENT IN A, LOWER PART OF MANTISSA
*                     * IN B)
*
$SETP,RP,105227       * SET UP A LIST OF POINTERS
*                     * NOTE: $SETP REPLACES .SETP AS OF 1913
*
.PACK,RP,105230       * CONVERT SIGNED MANTISSA OF REAL INTO NORMALIZE REAL
*                     * FORMAT
.CFER,RP,105231       * MOVE 4 WORDS (COMPLEX TRANSFER)   (IN F AND SOME E
*                     * SERIES)
*
*                     * ..FCM, ..TCM, .BLE, AND .NGL ARE AS OF REV 1926
*
..FCM,RP,105232       * COMPLEMENT A REAL               (IN F SERIES ONLY)
..TCM,RP,105233       * NEGATE A DOUBLE REAL            (IN F SERIES ONLY)
.BLE,RP,105207        * CONVERT REAL TO DOUBLE REAL     (IN F SERIES ONLY)
.NGL,RP,105214        * CONVERT DOUBLE REAL TO REAL     (IN F SERIES ONLY)
*
*     ***** EMA ENTRY POINTS (F AND E SERIES ONLY) *****
*
.EMAP,RP,105257       * RESOLVE REFERENCES TO EMA ELEMENTS
.EMIO,RP,105240       * USED FOR I/O FROM EMA ARRAYS
MMAP,RP,105241        * MAPS PHYSICAL PAGES INTO LOGICAL ADDRESS SPACE
*
*     ***** 3-WORD ENTRY POINTS (IN F SERIES ONLY) *****
*
.XADD,RP,105001       * EXTENDED REAL ADDITION
.XSUB,RP,105021       * EXTENDED REAL SUBTRACTION
.XMPY,RP,105041       * EXTENDED REAL MULTIPLCATION
.XDIV,RP,105061       * EXTENDED REAL DIVISION
.XFXS,RP,105101       * EXTENDED REAL TO INTEGER FIX
.DINT,RP,105101       * EXTENDED REAL TO INTEGER FIX (NOTE .DINT FOR FTN
*                     * INTERFACE, SAME ENTRY POINT AS .XFXS)
.XFXD,RP,105105       * EXTENDED REAL TO DOUBLE INTEGER FIX
.XFTS,RP,105121       * INTEGER TO EXTENDED REAL FLOAT
.IDBL,RP,105121       * INTEGER TO EXTENDED REAL FLOAT  (NOTE: FTN INTERFACE
*                     * SAME ENTRY POINT AS .XFTS)
.XFTD,RP,105125       * DOUBLE INTEGER TO EXTENDED REAL FLOAT
*
*     ***** 4-WORD ENTRY POINTS (IF F SERIES ONLY) *****
*
.TADD,RP,105002       * DOUBLE REAL ADDITION
.TSUB,RP,105022       * DOUBLE REAL SUBTRACTION
.TMPY,RP,105042       * DOUBLE REAL MULTIPLY
.TDIV,RP,105062       * DOUBLE REAL DIVIDE
.TFXS,RP,105102       * DOUBLE REAL TO INTEGER FIX
.TINT,RP,105102       * DOUBLE REAL TO INTEGER FIX  (NOTE: FTN INTERFACE,
*                     * SAME ENTRY POINT AS .TFXS)
.TFXD,RP,105106       * DOUBLE REAL TO DOUBLE INTEGER FIX
.TFTS,RP,105122       * INTEGER TO DOUBLE REAL FLOAT
.ITBL,RP,105122       * INTEGER TO DOUBLE REAL FLOAT (NOTE: FTN INTERFACE,
*                     * SAME ENTRY POINT AS .TFTS)
.TFTD,RP,105126       * DOUBLE INTEGER TO DOUBLE REAL FLOAT
*
```

```
*      ***** DOUBLE INTEGER ENTRY POINTS (FFP) (IN F SERIES ONLY) *****
*
.DAD,RP,105014       * DOUBLE INTEGER ADDITION
.DSB,RP,105034       * DOUBLE INTEGER SUBTRACTION
.DMP,RP,105054       * DOUBLE INTEGER MULTIPLICATION
.DDI,RP,105074       * DOUBLE INTEGER DIVISION
.DSBR,RP,105114      * DOUBLE INTEGER SUBTRACTION (REVERSED)
.DDIR,RP,105134      * DOUBLE INTEGER DIVISION (REVERSED)
.DNG,RP,105203       * DOUBLE INTEGER NEGATE
.DIN,RP,105210       * DOUBLE INTEGER INCREMENT
.DDE,RP,105211       * DOUBLE INTEGER DECREMENT
.DIS,RP,105212       * DOUBLE INTEGER INCREMENT AND SKIP IF 0
.DDS,RP,105213       * DOUBLE INTEGER DECREMENT AND SKIP IF 0
.DCO,RP,105204       * DOUBLE INTEGER COMPARE
*
*      ***** SIS ENTRY POINTS (IN F SERIES ONLY) *****
*
TAN,RP,105320        * TANGENT
SQRT,RP,105321       * SQUARE ROOT
ALOG,RP,105322       * NATURAL LOGARITHM LN(X)
ATAN,RP,105323       * ARCTANGENT
COS,RP,105324        * COSINE
SIN,RP,105325        * SINE
EXP,RP,105326        * EXPONENTIAL E**X
ALOGT,RP,105327      * LOGARITHM LOG10(X)
TANH,RP,105330       * HYPERBOLIC TANGENT
*
TRNL,RP,105331       * EVALUATE THE QUOTIENT OF 2 POLYNOMIALS IN DOUBLE
*                    * PRECISION
DPOLY,RP,105331      * EVALUATE THE QUOTIENT OF 2 POLYNOMIALS IN DOUBLE
*                    * PRECISION
*                    * NOTE: DPOLY REPLACES TRNL AS OF 1926 (SAME ROUTINE)
*                    * DPOLY IS USED IN OTHER SUBROUTINES SUCH AS DCOS
*                    * AND DSIN
*
*                    * /CMRT, /ATLG, .FPWR, AND .TPWR ARE AS OF REV 1926
*
/CMRT,RP,105332      * RANGE REDUCTION FUNCTION
/ATLG,RP,105333      * COMPUTE (1-X)/(1+X) IN DOUBLE PRECISION
.FPWR,RP,105334      * COMPUTE X**I FOR REAL X AND UNSIGNED INTEGER I
.TPWR,RP,105335      * COMPUTE X**I FOR DOUBLE REAL X AND UNSIGNED
*                    * INTEGER I
*
*      ***** VIS ENTRY POINTS (IN F SERIES ONLY) *****
*
.VECT,RP,101460      * FIRST OF TWO WORDS (USED BY SOFTWARE IN %VLIB TO
*                    * GET TO TWO WORD OPCODES)
VPIV,RP,101461       * PIVOT ROUTINE
VABS,RP,101462       * ABSOLUTE VALUE ROUTINE
VSUM,RP,101463       * SUM THE ARRAY ELEMENTS
VNRM,RP,101464       * SUM THE ABSOLUTE VALUE OF THE ELEMENTS
VDOT,RP,101465       * DOT PRODUCT ROUTINE
VMAX,RP,101466       * FIND THE LARGEST ARRAY ELEMENT
VMAB,RP,101467       * FIND THE LARGEST ARRAY ELEMENT (ABSOLUTE VALUE)
VMIN,RP,101470       * FIND THE SMALLEST ARRAY ELEMENT
VMIB,RP,101471       * FIND THE SMALLEST ARRAY ELEMENT (ABSOLUTE VALUE)
VMOV,RP,101472       * COPY AN ARRAY INTO AN OTHER ARRAY
VSWP,RP,101473       * EXCHANGE ELEMENTS OF TWO ARRAYS
.ERES,RP,101474      * CALCULATES 2 WORD OFFSET FOR EMA ARRAY ELEMENTS
.VSET,RP,101476      * CALCULATES MAP TABLE FORM .ERES INFORMATION
.ESEG,RP,101475      * PREFORMS THE MAPPING FROM THE MAP TABLE FOUND WITH
*                    * .VSET
.DVCT,RP,105460      * FIRST OF TWO WORDS (USED BY SOFTWARE IN %VLIB TO
*                    * GET TO TWO WORD OPCODES)
DVPIV,RP,105461      * PIVOT ROUTINE FOR DOUBLE REAL ARRAYS
DVABS,RP,105462      * ABSOLUTE VALUE ROUTINE FOR DOUBLE REAL ARRAYS
DVSUM,RP,105463      * SUM THE ARRAY ELEMENTS FOR DOUBLE REAL ARRAYS
DVNRM,RP,105464      * SUM THE ABSOLUTE VALUE OF THE ELEMENTS IN A DOUBLE
*                    * REAL ARRAY
DVDOT,RP,105465      * DOT PRODUCT ROUTINE FOR DOUBLE REAL ARRAYS
DVMAX,RP,105466      * FIND THE LARGEST ARRAY ELEMENT IN A DOUBLE REAL
*                    * ARRAY
```

```
DVMAB,RP,105467      * FIND THE LARGEST ARRAY ELEMENT IN A DOUBLE REAL
*                    * ARRAY (ABSOLUTE VALUE)
DVMIN,RP,105470      * FIND THE SMALLEST ARRAY ELEMENT IN A DOUBLE REAL
*                    * ARRAY
DVMIB,RP,105471      * FIND THE SMALLEST ARRAY ELEMENT IN A DOUBLE REAL
*                    * ARRAY (ABSOLUTE VALUE)
DVMOV,RP,105472      * COPY A DOUBLE REAL ARRAY INTO ANOTHER DOUBLE REAL
*                    * ARRAY
DVSWP,RP,105473      * EXCHANGE ELEMENTS OF TWO DOUBLE REAL ARRAYS
*
```

To clarify some points which may cause confusion:

- The routines .XADD, .XSUB, .XMPY, and .XDIV in the E and M series have different microcode addresses in the F series.
- The routines XADD, XSUB, XMPY and XDIV are FTN interfaces in the E and M series, but don't exist in the F series. Their microcode addresses are used for other modules in the F series, e.g., XADD in the E has microcode address for .BLE in the F.
- In the F series .DINT is the same as .XFXS. .DINT is the routine called by the FTN compiler when it generates code, while .XFXS conforms to the nomenclature used by other microcode entry points. The same applies to .IDBL vs. .XFTS, .TINT vs. TXFS and .ITBL vs. .TFTS.

In general, the nomenclature is as follows:

X — extended real
T — double real
S — single integer
D — double integer

- Not all of the VIS microcode can be directly RP'd in the generation. (The generator cannot handle 2 word RP's). To get around this problem, a software interface to get to the 2 word microcode opcodes was developed; it is called %VLIB. %VLIB is relocated during generation and allows the access of such modules as VADD, VSUB, VMPY, etc.
- The routine .CFER exists in all F series microcode and all E's with FFP microcode part numbers of:

13306-80013
13306-80014
13306-80015
13306-80016
13306-80017
13306-80018
5090-0589
5090-0590
5090-5091

BIT BUCKET

USER CODES AS AN AID TO DISC HOUSEKEEPING

Ed DeMers/Consultant
El Cajon, CA

INTRODUCTION

At some time almost every disc based system user faces the problem of tidying up the disc after prolonged use. Typically the issues in question are:

- What is the purpose of this file?
- Who created it?
- Should it be saved or purged?

A system of user codes can be an aid to answering these questions. Additionally, the system permits collections of file names with similar characteristics to be grouped, and may help answer the question, "How did all those megabytes disappear?"

BACKGROUND

The File Manager directory, as it is written on the disk, contains a word that is used for type 2 and type 6 files to contain the record length parameter. This word is reserved in the directory for all types of files, but is not used for types 3, 4, or 5 (their record lengths are variable). Depending on the parameters specified in the "CREAT" command for file types 3, 4, or 5, this word may be zero, or some meaningless random value. The User Code (UCODE) as discussed here, is a word containing two characters, which is stored in the location reserved for record length, and may be used as a parameter in any of the File Manager commands.

ADDING A UCODE

As seen from the above discussion, if a call to CREAT is being used to create a type 3, 4, or 5 file, the "ISIZE" array may be defined with the first word specifying the size of the file to be created (in blocks), and the second word specifying the value of some ASCII character pair to be added to the directory. The format of the CREAT call appears below for reference:

```
CALL CREAT(IDCB,IERR,NAME,ISIZE,ITYPE,ISECU,ICR,IDCBS)
```

IDCB	Data control block
IERR	Error return
NAME	File Name
ISIZE	File size; a 2-word array with the number of blocks in the first word, and, for type 2 files, the record length in the second word
ITYPE	File type
ISECU	Security code
ICR	Cartridge reference
IDCBS	DCB buffer size

The FMP will automatically enter the information above in the appropriate words of the directory. As an alternative, a program has been written using the File Manager commands to insert into the directory the desired character pair in the position reserved for record length. The program UCODE interactively requests the User Code and file name from the user, and inserts this user code in the proper word of the directory. There is also a place in the namr parameter to indicate a record size, (the fifth subparameter), which can be filled with the ASCII equivalent of the desired character pair when doing file manipulations.

ASSIGNMENT OF THE CODES

It must fall to some authority to assign the codes to be used. The codes can be related to the user's name, project, or some other identifier. It might seem that being limited to two characters is rather restrictive, but experience has shown that this is not the case. To supplement the alphabetic character set, most of the special characters may be used (with the exceptions of the comma, colon, minus, and the plus), as well as the digits 0 through 9. The lower case characters may also be used when the interactive console has this capability.

LISTING THE FILES

A program has been written to conveniently list file names in a specified directory, and their associated User Codes (if they are present). The listing format is arranged to display five file names across an 80 column page, or eight file names across a 120 column page. Any given file name is listed only once, (extents are ignored). The calling parameters of the routine ULIST also allow listing of only those files having a given User Code. Occasionally there may be large quantities of file names on a given cartridge, and it may be desirable to list the name of one that is near the end of the list. The calling parameter "SKIP" permits listings to begin with the n^th^ entry in the directory, so that the desired item may be found more easily.

The calling sequence to ULIST is as follows:

```
CALL ULIST ( LU, CART, UCODE, LISTLU, SKIP )
```

LU	The interactive console device
CART	The cartridge number or its LU
UCODE	The desired Ucodes to be selected
LISTLU	The device on which the list is to be displayed
SKIP	An integer defining the number of names to be skipped before listing is to begin.

CART	Defaults to a built in value (installation dependent)
UCODE	Defaults to all Ucodes
LISTLU	Defaults to the interactive LU
SKIP	Defaults to no skip

As ULIST "plows" through the directory, it counts the number of directory slots that have been used, the number that have random or unidentified values, and the number that are available for use. (In this way, a FMGR-14 error (directory full) may be predicted.) The number of files that are not of types 3, 4 or 5, and the number of files that do not have User Codes specified are also counted. Finally, as an additional feature, the list is dated to aid in record keeping.

SORTING FEATURE

If a sorted list is of interest to the user, the program "ALIST" will sort the file names according to the ASCII equivalent of the User Code. The calling sequence for ALIST is identical to that of ULIST, except that "SKIP" has been replaced with an ASCII character "S". If "S" is specified, the only files which are sorted are those of type 3 or type 4. Hence, only the source files, and none of the relocatable (type 5) files are listed. If the "S" is not specified, the sort considers file types 3, 4, and 5. The ALIST parameter, "LISTLU" will vary the width of the output listing. If 6 is specified here, a 132 column format will be used. This format may be easily changed by the user.

WHERE DID THE SPACE GO TO?

An alternate version of the program ALIST, is the program BLIST. BLIST makes no use of User Codes, but instead sorts the files in order of their block size. Note that BLIST will consider primary entries, and also each extent of a file. Both ALIST and BLIST use a "BUBBLE" sort, and therefore run rather slowly when the list is large.

Calling sequence:

```
CALL BLIST ( LU, CART, LISTLU, MAXSIZE )
```

LU	The interactive console
CART	The cartridge number or LU (negative)
LISTLU	The LU where the list is displayed
MAXSIZE	The largest size block to be sorted

CART	Defaults to a built in value (installation dependent)
LISTLU	Defaults to LU
MAXSIZE	Defaults to list all

Another useful program is "NLIST" which lists all files without a recognizable User Code.

Calling sequence:

```
RU,NLIST,(LU, CART, FFLAG, LISTLU)
```

If FFLAG is set to character A, all files will be checked. FFLAG defaults to type 3 and 4 files only.

EXTENSIONS TO THE SYSTEM

The program "UFILE" is a useful extension to the User Code system, and provides the user with the ability to create a new file containing a list of all file names that have a given User Code.

Calling sequence:

```
:RU,UFILE,LU,CR,UCODE,UFILE,SC,CR2
```

LU	Interactive device for operator communication
CR	Cartridge number or LU (negative) which is to be searched
UCODE	A 2 character User Code which defines the Files
UFILE	The file on which the list is to be written
SC	Security code of the created file
CR2	Cartridge on which UFILE is to be found or created

LU	Defaults to 1
CR	Defaults to cartridge 9
UCODE	If defaulted, will be requested interactively
UFILE	Defaults to a file name UCFILE
SC	Defaults to no security code
CR2	Defaults to CR

Editor's Note:

The above programs have been made available to our readers through LOCUS as the UCU group with catalog number 22683-13381.

IMPROVED DVA05/MODEM/2645 PERFORMANCE

Richard Raskin/HP Neely Sales Region

A customer recently complained about the performance he was getting from a 2645 terminal which was being driven by DVA05, via modems, over a leased line. An on site visit, complete with a C.E. communications specialist, a few scopes, and various other pieces of test equipment revealed that both the CPU and the leased line were behaving up to speed.

After some careful observation, it was suggested that the ENQ/ACK overhead was causing the problem. Since the customer was using block mode read/write requests, the DVA05 transmission block size was increased from the standard 33 character maximum. This modification resulted in a significant performance improvement.

A few words of caution are in order. First, performance degradation will occur whenever the length of the read/write requests are greater than the DVA05 transmission block size. The greater the difference, the more often the ENQ/ACK sequence will be performed. This is most common with block mode requests such as the particular situation we encountered, where the user block sizes ranged from 200 to 300 words. With an overhead of 50 to 100 milliseconds per ENQ/ACK sequence, the performance tended to degrade as the request size increased.

The second note of caution involves modification of the DVA05 transmission block size. The 2645 terminal has a hardware buffer of slightly less than 100 characters. When increasing the transmission block size, allowance must be made for terminal control functions such as cursor positioning commands. Many terminal control functions require pad or filler characters (i.e. nulls) to allow for the completion of the command by the terminal hardware. These filler characters occupy buffer space. Failure to allow adequate space for filler characters (DVA05 tramsmission block size set too high) will result in buffer overruns, lost data, etc.

The final caution note concerns the method used to modify the DVA05 transmission block size. While this can be done by patching the driver using CMMx (where x is M, 3, 4, etc.), it is best to modify the driver source and then perform a system generation. It seems that the constant that DVA05 uses to initialize the character count is also used for some other operations, such as supression of the CR/LF if the last character is an underline.

IN SUMMARY

1. Evaluate each situation indivudually and correct any other problems.
2. Verify that a change to transmission block size will improve performance by temporarily patching DVA05. A block size of 80 characters (177660 octal) is recommended.
3. Permanently modify DVA05 source code and include the new relocatable module in the next system generation.

RECOVERING SYSTEM RESOURCES VIA AUTOMATIC TIMED REBOOT

Glenn Talbott/HP Data Systems Division

As the system manager of a busy RTE-IV system, I recently found that more and more of my time was taken up recovering lost resources, especially ID segments and system tracks. The system I manage is an RTE-IVA in an E-Series Computer with 416K bytes of high speed memory. Included in the system are a 7905 disc, a 7920 disc, DS/1000, and seventeen 264X terminals on 12966 interfaces. The primary reason for loss of system resources was the forty to fifty users on the terminals.

The system was installed in Hewlett-Packard's Data Systems Division Technical Marketing department for utility and demonstration program development, manual writing, and training. With this utilization, for most of the working day the system would average around fifteen people writing and debugging programs, or using various word processing programs to write manuals.

If I booted up the system on Monday morning, by Wednesday afternoon user program activity would grind to a halt. The reason for this slowdown was that too many of the users left too many temporary programs loaded so that the system either ran out of ID segments or system tracks. If the lack of ID segments was the problem, users were unable to run programs from type 6 files. If the lack of system tracks was the problem, the Loader or Editor would suspend waiting for disc space, or swapping would cease, and scheduled programs would be waiting for partitions. In either case there would be an inconvenient slow down or stoppage of system use.

In some cases I was able to 'off' programs until the system started up again, other times it was easier to just reboot the system. Obviously, a method of preventing this system slowdown from happening had to be found. Asking the users to 'off' their programs when finished was one course of action, but it was easy to forget. I tried arriving at work early enough to reboot the system every morning, but invariably someone would already be there.

Then I found the solution. Using a combination of hardware and software components, I was able to set up a method of automatically rebooting the system at 2 AM every day. The method as described below is quite easy to implement and I would recommend it for anyone else with similar problems.

First the hardware; the E-Series Computer Operating and Reference manual (02109-90014) describes Remote Program Loading (RPL) in section 2-30. In this application the computer will automatically reboot if the LOCK/OPERATE switch is in the LOCK position, the RPL Configuration switches on the CPU board are set correctly, and a programmed halt instruction is executed, octal code 1060xx. (A halt is normally considered as an octal 1020xx, but if bit 11 is set the RPL feature is enabled.) The manual implies that this method is used for booting from a communications interface. However if an RPL compatible disc loader ROM is installed it is just as easy to boot off of the disc.

The best description of setting the RPL configuration switches and the RPL compatible disc loader ROMs is in the HP 12992 Loader ROM's Installation Manual (12992-90001). The RPL compatible disc loader ROMs are as follows:

12992B	7905/7906/7920 Disc Loader
12992E	9885 Flexible Disc Loader
12992F	7900/7901 Disc Loader

The RPL configuration switches (U173S1 through S8) are set as follows for the 7905/06/20 disc:

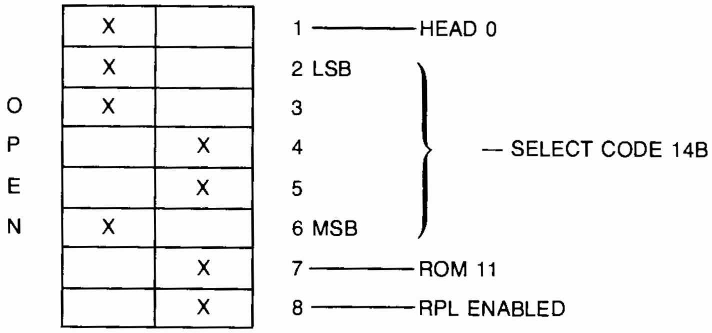
S1 open — select head 0
closed — select head 1

S2-S6 octal select code of disc interface, closed = 1

S7 open — select ROM position 10
closed — select ROM position 11
(note for RPL, ROMs may only be installed in 10 or 11)

S8 closed — enable RPL

The following is an RPL Configuration Switch setting example:



	X		1	HEAD 0
	X		2 LSB	
O	X		3	
P		X	4	— SELECT CODE 14B
E		X	5	
N	X		6 MSB	
		X	7	ROM 11
		X	8	RPL ENABLED

Remember to leave the computer's LOCK/OPERATE switch in LOCK for automatic reboot.

The software involves a program to execute the 1060xx halt and, in the 'WELCOM' file, scheduling the program to run at 2:00 AM every day. The program must include a call to $LIBR to go privileged prior to executing the 1060xx halt. Note that the mnemonic 'HLT' will not work, the halt must be entered as OCT 1060xx. The program could do any clean-up your system requires or print out warning messages to all the terminals, however I chose to keep mine as simple as possible as can be seen in the following listing.

```
                  ASMB,Q
00000                   NAM HALT,3,20 HALTS THE CPU WITH A 106077
                        EXT $LIBR
00000 000000      HALT  NOP
00001 000001X           JSB $LIBR
00002 000000            NOP
00003 106077            OCT 106077
                        END HALT
```

Scheduling the program is accomplished in the WELCOM file using the SYIT and SYON commands as follows:

```
   .
   .
   .
:SYIT,HALT,4,0,2,0,0,0
:SYONIH,HALT
   .
   .
   .
```

The 4 in the SYIT parameters says run every hour, but the following 0 says run only once. The 2,0,0,0 says run first at 2:00:00.00. The IH in the SYONIH says inhibit passing the run string. There is no sense in having the run string tie up a block of SAM for 24 hours. This procedure worked fine, and the only remaining problem was how to set the system time correctly at 2 AM. The method I chose was to get the system time from another node in our DS/1000 network. I wrote the following program to accomplish this, and other systems in the network are now using it to maintain a standard network time even though they don't reboot every day.

```
FTN4,L
      PROGRAM DDATE(19,40),SET DATE AND TIME USING DS/1000
C
C THIS WAS IMPLEMENTED TO FACILITATE AUTOMATIC BOOT UP AT
C 2 AM EVERY DAY
C
C BY GLENN TALBOTT  3/11/79
C
C
      INTEGER A,B,TIME(21),NODE(5),DSER(2),DTIME(5),DYEAR,DCNT,CRT
      EQUIVALENCE (A,DSER),(B,DSER(2))
C
      DATA TIME/2HTM,2H, /
C
C  THIS PROGRAM REQUEST'S THE DATE AND TIME USING A DEXEC 11
C  CALL TO THE NODE PASSED IN THE FIRST PARAMETER.
C  THE PROGRAM THEN FORMATS BOTH INTO A SINGLE MESSAGE
C  OF THE FORM  TM,YEAR,DAY,HOURS,MINUTES,SECONDS AND PASSES
C  IT TO THE SYSTEM THROUGH THE MESSAGE PROCESSOR.
C
      CALL RMPAR (NODE)
      CRT = LOGLU(IDUMY)
      TIME (6) = 2H,
      TIME (10) = 2H,
      TIME (14) = 2H,
      TIME (18) = 2H,
C
C GET THE DS/1000 TIME FROM NODE
C
      CALL DEXEC(NODE,11+100000B,DTIME,DYEAR)
      GO TO 800
C
C NORMAL RETURN
C
802   CALL CNUMD(DYEAR,TIME(3))
      CALL CNUMD(DTIME(5),TIME(7))
      CALL CNUMD(DTIME(4),TIME(11))
      CALL CNUMD(DTIME(3),TIME(15))
      CALL CNUMD(DTIME(2),TIME(19))
      DCNT=21
      GO TO 801
C
C DS/1000 ERROR RETURN, PRINT ERROR AND RETURN
C
800   CALL ABREG(A,B)
      CALL EXEC(2,CRT,DSER,2)
      STOP
C
C NOW SET THE SYSTEM TIME
C
801   DCNT=2*DCNT
      I = MESSS (TIME,DCNT,CRT)
C
C IF ERROR I WILL BE NE 0
C
      IF (I.NE.0) CALL EXEC (2,CRT,TIME,I)
      IF (I.NE.0) STOP
```

```
C
C NOW GET AND PRINT SYSTEM TIME
C
      CALL FTIME(TIME)
      CALL EXEC(2,CRT,TIME,15)
      END
      END$
```

This program is also scheduled from the WELCOM file. It must come after DS/1000 initialization, and then the DS monitor UPLIN must be rescheduled along with anything else in the time list. The portion of the WELCOM file that does this is as follows:

```
   .
   .
   .
:RU,LSTEN,*L,ST,EN
:RU,DDATE,200
:SYONIH,UPLIN,NOW
   .
   .
   .
```

An alternate method of setting the system time would be to use a 59309A HPIB Digital Clock and a program to read the time from it. This method is described in Hewlett-Packard Application Note 401-8 and the program is available in the contributed library.

Using the procedure described in this article I have been able to keep the system up during high usage periods. With the periodic cleanup provided by rebooting every day there has been a noticeable improvement in system performance.

With the introduction of HP's RTE-IVB Operating system with Session Monitor the requirement for this reboot procedure will be reduced significantly. The new Session Monitor will automatically off all programs temporarily loaded under a session when the session is ended. This will return the ID segments and disc tracks used by those programs. However, the reboot procedure described here will still be of value in recovering lost class numbers, resource numbers and fragmented SAM.

While this procedure may not be useful to everyone, I would recomend it to anyone who has a fixed time during non-working hours when the system could be rebooted without inconveniencing any users. Your system may not appear to need this periodic cleanup, but it probably would benefit from it anyway.

OPERATIONS MANAGEMENT

SCHEDULING DATA ACQUISITION PROGRAMS WITHOUT PROVIDING GENERAL SYSTEM ACCESS TO TERMINAL USERS

Frank K. Fulton/Research and Development
Computer Science Services
R.J. Reynolds Tobacco Company
Winston Salem, North Carolina 27102

Consider the following situation: there are one or more applications on your system that call for routine data entry from non-technical personnel (as is often required in inspection and testing of manufactured goods). A number of interactive terminals (CRTs and keyboard printers) are available which can be used to schedule different data acquisition programs. The choice of data acquisition programs available to any one terminal will need to be displayed as a menu which will be printed when any key on the terminal is pressed. Generally, each terminal will be dedicated to some specific set of tasks and the variety of menus offered will be less than the number of terminals.

THE REQUIREMENTS

It is desired that each terminal be able to schedule its data acquisition programs without interference from other terminals (which use the same programs). It is hoped that something can be done to minimize the number of ID segments used since the number of data acquisition programs in use at any one time may be considerably less than the total number of data acquisition programs (menu items) available to all terminals.

RULING OUT POSSIBILITIES

The multi-terminal monitor can handle most of the requirements, but one drawback is that it allows users to enter file manager and system commands. In this instance, such access is undesirable. Another problem is that the 'HI' file schedules 'with wait', and any read issued to the user terminal by the scheduled program (such as for a response to a menu) prevents access to the 'HI' file by another terminal. Let's also rule out an MTM-like terminal handler (as described in Gary McCarney's article in vol. III, issue 2 of the Communicator) on the basis of the need for simplicity and maintainability of the data acquisition programs.

A menu can be sent to a terminal by directly scheduling a menu selection program from an interrupt. If this is done by entering the program's name in the interrupt table of the system generation, some problems arise:

1. The copies of the menu selection program must be generated into the system and, if the program has appended subroutines, they must be renamed in each copy to avoid a generation error 08 (duplicate program name).

2. As an alternative, dummy copies of the programs may be generated into the system, and later replaced on line (a cumbersome practice).

3. In either case, maintaining several copies of identical (or worse, almost identical) programs can be a nuisance.

A BETTER APPROACH

The approach presented here takes the following form:

SYSTEM GENERATION CONSIDERATIONS

```
      .
      .
      .
REL,%TERML
      .
      .
      .
TERML,1,50
      .
      .
      .
'SC',PRG,TERML
'SC',PRG,TERML
'SC',PRG,TERML
```

SCHEDULING SEQUENCE USED TO ACTIVATE DATA INPUT

1. A terminal key is pressed causing ...
2. A memory resident program 'TERML' to be scheduled. It is turn schedules the proper copy of the menu selection program 'SCHED' (renamed for the logical unit number of the scheduling terminal).
3. In response to the menu offered, the terminal operator picks an item and the proper data acquisition program is scheduled.
4. Following execution of the data acquisition program, its ID segment is removed.

Hitting a key on a terminal causes a short memory resident program 'TERML' to schedule the proper copy of a menu selection program 'SCH..' (where .. is the logical unit number of the terminal). For example: a key pressed on LU 9 would result in 'SCH09' being scheduled. A listing of 'TERML' is given below. The program's purpose is to minimize the possibility of it being busy when a key pressed on another terminal causes 'TERML' to again be scheduled. This requirement is met by the fact that 'TERML' is memory resident, short, and of high priority. 'SCH..' could be a data acquisition program, but maintenance can be simplified by making 'SCH..' a menu selection program which can schedule any data acquisition program. Only a single menu selection program is needed, since each terminal uses a renamed copy. The program will select the proper menu for the terminal by testing the terminal's logical unit number.

```
ASMB,R,L
      NAM TERML,1,50
*
*    THIS IS A CORE RESIDENT PROGRAM TO SCHEDULE THE PROPER
*    COPY OF 'SCHED'.
*
      ENT TERML
      EXT EQLU,EXEC
*
TERML NOP
*
      JSB EQLU         B-REG CONTAINED INTERRUPT EQT
      DEF *+2          SO THIS WILL GET THE LU.
      DEF LU
*
      CLB              CONVERT THE LU TO ASCII:
      LDA LU
      DIV D10
      STA B
      ADB B60          GETS FIRST ASCII CHARACTER
      STB TENS
      MPY D10
      CMA,INA
      ADA LU
      ADA B60          SECOND ASCII DIGIT
      STA ONES
*
```

```
       LDB ANAM2     LOAD BYTES WITH NEW CHARACTERS:
       RBL           GET BYTE ADDRESS...
       INB           REPLACE BEGINNING WITH 2ND WORD, 2ND CHAR
       LDA TENS      OF 'SCH..' NAME
       SBT
       LDA ONES
       SBT
*
       JSB EXEC      SCHEDULE MENU SELECTION PROGRAM
       DEF *+4
       DEF D10       WITHOUT WAIT
       DEF NAM
       DEF LU        AND SEND LU NUMBER
*
       JSB EXEC      EXIT
       DEF *+2
       DEF D6
*
B      EQU 1B        B IS B-REG
*
D6     DEC 6
D10    DEC 10
B60    OCT 60
LU     BSS 1
ONES   BSS 1
TENS   BSS 1
ANAM2  DEF NAM+1     (LOOK AT 2ND WORD OF NAME)
NAM    ASC 3,SCH..
*
       END TERML
       END$
```

DUPLICATING AN ID

After a menu item is selected by the user, the menu selection program will need to duplicate an ID segment so that a copy of the selected data acquisition program can be used for the scheduling terminal. Following completion of that task, the copy will need to be programmatically 'offed' to release the ID segment back to the system. These features are available to users through batch monitor library routines 'IDDUP' (to duplicate an ID segment) and 'IDRPD' (to remove it). Briefly, ID segments can be duplicated if they satisfy the following conditions:

1. The program has been saved on logical unit 2 or 3 with the file manager 'SP' command.

2. An ID segment for that program resides in memory. Note: on boot-up, temporary program's IDs must be restored through use of the 'RP' command (in the welcome file) if they are to be duplicated later.

Errors for these routines are given in table 1.

An outline of the schedule program will look like this:

```
FTN,L
      PROGRAM SCHED
C
C   (MENU SELECTION PROGRAM)
C   COPIES OF THIS PROGRAM ARE USED TO OUTPUT ONE OF SEVERAL POSSIBLE
C   MENUS TO A TERMINAL AND ON REPLY FROM THE USER, SCHEDULE THE
C   CORRESPONDING DATA ACQUISITION PROGRAM:  APROG, BPROG, ETC.
C
      DIMENSION LU(5),IAB(2),NAM(3)
      DIMENSION NAMP1(3),NAMP2(3),NAMP3(3),...
C
      EQUIVALENCE (REG,IAB)
C
C   THESE ARE NAMES OF DATA ACQUISITION PROGRAMS.
C
      DATA NAMP1 / 2HAP,2HRO,2HG  /, NAMP2 / 2HBP,2HRO,2HG  / ...
C
C   PICK UP LU FROM 'TERML'.
C
      CALL RMPAR (LU)
C
C   DISARM TERMINAL FOR NOW.
C
      CALL EXEC (3,2100B)
C
C   IF THIS IS THE FIRST GROUP OF TERMINALS (E.G., 21,22, OR 23),
C   SEND THEM THIS MENU.  ELSE, GO TO TEST FOR NEXT MENU.
C
      IF ((LU.NE.21).AND.(LU.NE.22).AND.(LU.NE.23)) GO TO 50
C
C   OUTPUT FIRST TYPE OF MENU.
C
      WRITE (LU,5)
    5 FORMAT (/," SELECT PROCEDURE:",/,5X,"1 FOR TEST A",/,5X,
     1   "2 FOR TEST B",/,5X,"3 FOR TEST C")
C
C   READ AND SCHEDULE PROPER PROGRAM.
C
      READ (LU,*) IWANT
      GO TO (10,20,30), IWANT
C
   10 DO 15 I=1,3                     /RENAME COPY:
   15 NAM(I) = NAMP1(I)               /GET ORIGINAL PROGRAM NAME
      CALL QQ (LU,NAM)                /MAKE COPY NAME
      CALL IDDUP (NAMP1,NAM,IER)      /DUPLICATE ID
      IF ((IER.NE.0).AND.(IER.NE.23)) GO TO ...
                                      /GO TO ERROR IF CAN'T MAKE COPY
      REG = EXEC(9,NAM,LU)            /SCHEDULE COPY WITH WAIT (PASS LU)
      IF (IAB.NE.0) GO TO ...         /ERROR IF PROGRAM NOT DORMANT
                                       (IAB IS A-REG RETURNED.  IF PROGRAM
                                        WAS NOT SCHEDULED, NON-ZERO VALUE
                                        IS PROGRAM STATUS ON SCHED ATTEMPT)
      CALL IDRPD (NAM,IER)            /REMOVE ID ON PROGRAM TERMINATION
      IF (IER.NE.0) GO TO ...         /ERROR IF FAILURE TO REMOVE ID
      CALL EXEC (2,2000B)             /RE-ARM TERMINAL
      CALL EXEC (6)                   /DONE!
```

```
C
C   DO SAME FOR OTHER DATA ACQUISITION PROGRAMS ON THIS MENU.
C
   20 DO 25 I=1,3
   25 NAM(I) = NAMP2(I)
      CALL QQ (LU,NAM)
      CALL IDDUP (NAMP2,NAM,IER)
      IF ((IER.NE.0).AND.(IER.NE.23)) GO TO ...
      REG = EXEC(9,NAM,LU)
      IF (IAB.NE.0) GO TO ...
      CALL IDRPD (NAM,IER)
      IF (IER.NE.0) GO TO ...
      CALL EXEC (2,2000B)
      CALL EXEC (6)
         .
         .
         .
C
C   ... AND OTHER MENUS FOR OTHER TERMINALS.
C
   50 IF ((LU.NE.24).AND.(LU.NE.25) ...
         .
         .
         .
      END
      SUBROUTINE QQ (IDEV,NAM)
C
C   CONVERT IDEV TO TWO ASCII CHARS, REPLACE ANY SPACE WITH ZERO,
C   AND PLACE RESULT IN NAM (RIGHT BYTE NAM(2), LEFT BYTE NAM(3))
C   TO RENAME A COPY OF THE PROGRAM TO BE RP'ED.
C
      DIMENSION NAM(1)
C
      CALL CODE
      WRITE (II,10) IDEV
   10 FORMAT (I2)
      IF (IAND(II,77400B).EQ.20000B) II = IAND(II,177B) + 30000B
C
C   MOVE TO NAM.
C
      NAM(2) = IAND(NAM(2),77400B) + IAND(II,77400B)/256
      NAM(3) = 40B + IAND(II,177B)*256
      END
      END$
```

Pressing a key on a terminal with logical unit 23 would result in the printing of this message:

```
SELECT TEST PROCEDURE:
    1 FOR TEST A
    2 FOR TEST B
    3 FOR TEST C
```

(Here the terminal would wait for a response.)

If a response of '2' is given, data acquisition program 'BPR23' (the copy of 'BPROG' for terminal 23) will be scheduled.

MAKING COPIES OF THE MENU SELECTION PROGRAM

On boot-up, you must make a copy of the menu selection program for each terminal. It's fairly easy to load the schedule program and make copies of it with the file manager 'SP' command. But, it's a lot easier and faster to use 'IDDUP' to make the necessary IDs by executing a short program from the welcome file. The following program demonstrates this for ten terminals with logical unit numbers 21, 22, 23, ... , 54:

After boot-up, the following welcome file should be executed:

Boot-up welcome file

```
:RP,APROG      /'RP' DATA ACQUISITION PROGRAMS
:RP,BPROG
:RP,CPROG
    .
    .
    .
:RP,SCHED      /'RP' MENU SELECTION PROGRAM
:RU,SCDUP      /DUPLICATE IDS FOR EACH TERMINAL
:OF,SCHED      /SAVE ONE ID BY OFFING ORIGINAL PROGRAM
:TR
```

```
FTN4,L
      PROGRAM SCDUP (3,100)
C
C   GIVEN A TABLE OF LOGICAL UNITS, MAKE ID SEGMENT COPIES
C   OF SCHED FOR THEM ON BOOT-UP.
C
      DIMENSION LUTAB(10),NAM(3),NAMSC(3),LU(5)
C
      DATA LUTAB / 21,22,23,24,25,27,28,52,53,54 /
      DATA NAMSC / 2HQS,2HCH,2HD  /
C
C
      NTERM = 10
C
      DO 50 I=1,NTERM
      DO 10 J=1,3
   10 NAM(J) = NAMSC(J)
      CALL QQ (LUTAB(I),NAM)
      CALL IDDUP (NAMSC,NAM,IER)
      IF ((IER.EQ.0).OR.(IER.EQ.23)) GO TO 50
      WRITE (1,15) IER,LUTAB(I)
   15 FORMAT ("SCDUP: IDDUP ERROR ",I3," ON LU ",I2)
   50 CONTINUE
C
      END
```

SOME VARIATIONS AND TIPS

1. Programs which run only at a single terminal can be 'RP'ed on line with another batch spool monitor goodie called 'IDRPL'. it is a little tricky to use. The sequence is:

   - open the file
   - 'RP' the program with 'IDRPL'
   - execute the program
   - close the file
   - release the ID segment with 'IDRPD'

The given order must be strictly followed. The code below would replace code in the earlier example of 'SCHED' beginning with the label '10':

```
                .
                .
                .
10 CALL OPEN (IDCB,IER,NAM)      /OPEN FILE NAM (NAM IS THE FILE AND
                                  THE PROGRAM NAME
   IF (IER.LT.0) GO TO ...       /ERROR IF CAN'T OPEN FILE
   CALL IDRPL (IDCB,IER,NAM)     /'RP' TYPE 6 FILE
   IF ((IER.NE.0).AND.(IER.NE.23)) GO TO ...
                                 /ERROR IF CAN'T 'RP' PROGRAM
   CALL EXEC (9,NAM,LU)          /SCHEDULE PROGRAM WITH WAIT
   CALL CLOSE (IDCB,IER)         /ON COMPLETION, CLOSE FILE
   IF (IER.LT.0) GO TO ...       /ERROR IF CAN'T CLOSE
   CALL IDRPD (NAM,IER)          /RELEASE ID SEGMENT
   IF (IER.LT.0) GO TO ...       /ERROR IF CAN'T RELEASE ID
   CALL EXEC (3,2000B)           /RE-ARM TERMINAL
   CALL EXEC (6)                 /DONE!
                .
                .
                .
```

2. Maintaining the schedule program is easy if there is only one. Once the program is compiled and loaded (loading will not affect the terminals since they are using renamed copies), the new copies of the schedule program may be installed with the following transfer file during a brief lull in the use of the terminals.

```
:OF,SCH..         /REMOVE OLD RENAMED COPY IDS
:OF,SCH..          (WHERE .. ARE LU NUMBERS IN NAMES
:OF,SCH..           OF COPIES)
    .
    .
    .
:RU,SCDUP
:TR
```

IN SUMMARY

The principle advantage of this technique is that the scheduling programs impose no constraints or requirements on the data acquisition programs. Often the only parameter passed will be the terminal logical unit number.

Other support programs such as the Editor or even the File Manager can be scheduled in the same manner to allow as much or as little system access as desired.

Table 1. Error Returns

PROGRAM	ERROR	MEANING
IDDUP (duplicate an ID segment)	0	Successful completion
	14	ID not found on system
	16	ID not type 2 or 3
	17	ID not RP'ed
	23	ID copy already exists in system
	−15	ID copy name illegal
IDRPL ('RP' a type 6 file)	0	Successful completion
	−11	File DCB not open
	14	No blank ID segments available
	−15	Illegal name
	16	File not on LU 2 or LU3
	19	Type 6 file not saved on this system
	23	Duplicate program name
IDRPD (remove an ID segment)	0	Successful completion
	9	ID segment not found
	17	ID segment not setup by 'RP' command
	18	Program not dormant

HP-IB AND THE SRQ

Pete Almeroth/Systems Engineer, HP Cleveland

USING SRQ TO SCHEDULE PROGRAMS

SRQ is a signal generated (asserted) by an HP-IB device to flag its need for attention. It is an indicator signaling that a predetermined event has occurred. The action taken is determined by the user. SRQ bypasses the normal handshaking protocol and is asserted on a separate line on the bus.

The SRQ is to the HP-IB controller (an interface card in the CPU), what an interrupt is to the computer, with one important exception. The interrupt to the computer has to be solicited before it is recognized. Unsolicited interrupts are ignored by the completion section of the driver assigned to process those interrupts. The SRQ, on the other hand, does not have to be solicited, and may occur at any time. A typical use for SRQ is to schedule an alarm program when some predetermined event occurs, such as an upper temperature limit being exceeded, or a maximum grind time being reached.

It is usually necessary to set a mask word local to the particular HP-IB device to specify what conditions (events) will cause the device to assert SRQ. Once the device asserts SRQ, the controller alerts the driver, (DVR37) which in turn does a serial poll of all the devices on the bus to determine which device caused the SRQ. All of the devices on the bus are polled sequentially. A device responds to the serial poll by sending a status byte back with bit 6 set to 1 if that device generated the SRQ. At this point the SRQ is reset. If two or more devices assert SRQ simultaneously, the device with the lower address will get serviced first. This address is equal to the sub-channel number defined in the DRT. When the driver sees the SRQ bit set in a status byte, it checks the 5 word EQT extension associated with this device for the name of the program linked to this SRQ. The program is scheduled via a call to $LIST. If no program name is linked, the SRQ is effectively ignored. This linkage in the EQT extension is established via a "CALL SRQ(LU,16,IPROG)" where LU is the logical unit number of the 'SRQ' device, and IPROG contains the name of the program to be scheduled and the length of that name (e.g., DATA IPROG/5,2HAL,2HAR,2HM/).

Before using SRQ, it is necessary to understand the device configuration word, maintained by the driver, that defines what action the bus controller takes in managing the device.

CONFIGURATION WORD

S	RDI	JOP	E00	000	000
15					0

The I-J-O-P bits are device dependent and describe the requirements for sending and receiving data. They are fully described in the HP-IB users manual (section 4) and are not relevent to this discussion.

The S-R-D-E bits require closer attention, and are the subject of the rest of this article.

The S bit is used to tell the controller whether or not to interrupt the current I/O operation on the bus when SRQ is asserted. If S is set to 0 the current I/O operation is completed before SRQ will cause the alarm program to be scheduled. If the I/O is very long or the device very slow, this delay in alarm program execution may be undesirable. If S is set to 1 when SRQ occurs, the I/O operation in progress is aborted, (not interrupted), the alarm program is scheduled, and an error condition is set.

What happens next depends on three things:

- the state of the R bit
- the state of the E bit
- whether the bus is buffered or unbuffered

The R bit is effective only if the S bit is set to 1. It is used to restart the current I/O operation automatically when SRQ is asserted. The R option must not be used if the restarted I/O operation will again assert SRQ. This will cause an infinite loop situation where the I/O operation is continually restarted. If the restarted I/O operation is completed successfully, control is returned to the user program. If the restart attempt fails, an error condition is set.

The E bit is used to define error handling. It is significant only if the bus is unbuffered. Errors are processed either by the operating system (if E is set to 0), or by the user's program (if E is set to 1). If E is set to 0 and an error condition occurs, the operating system downs the 'SRQ' device, outputs "I/O NR LUxx" to the console and puts the user's program into a wait state. The operator must "UP,eqt" to continue. If E is set to 1 and an error occurs, control is returned to user program and the error status is set to 3. Error status is available to the user via the function IBERR (e.g. IERR=IBERR(LU) where LU is the logical unit number of device on bus). If EQT time-out occurs and E is set to 1, the device will not be downed, but control returns to the user program and the error status is set to 1.

If the bus is buffered, error handling is identical to the case where the E bit is 0 as discussed above.

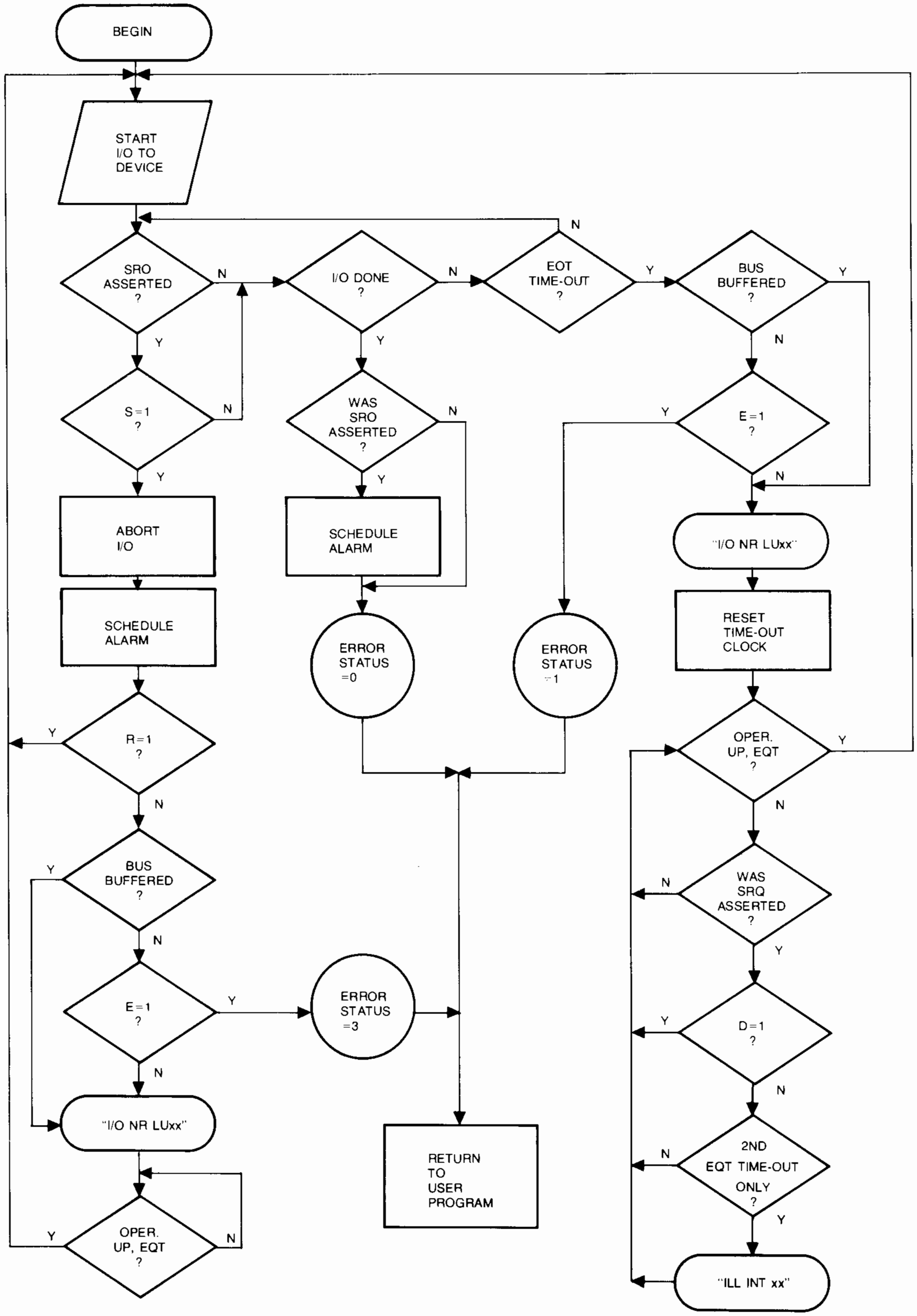
The D bit specifies a DMA operation and its use is straight forward. One word of caution is in order. The D bit must not be used in conjunction with the S bit, since it is not practical to abort I/O under DMA; thus scheduling of the alarm program is not accomplished. DMA operates on cycle stealing basis and is not under control of the CPU.

If it is desired to abort an I/O operation to schedule an alarm program and then return control to the user program without operator assistance, two methods may be used. The bus must be unbuffered and the D bit set to 0 in both cases.

1. If it is desired to restart the I/O automatically, and the I/O operation is not related to SRQ, the S and R bits should both be set to 1. When SRQ occurs, the I/O operation will be aborted, the alarm program will be scheduled, the I/O operation will be restarted and control will return to the user; when I/O is finished, the error status is set to 0.

2. If it is necessary to return to the user program before restarting I/O, as is the case when the I/O operation is directly related to SRQ, the S and E bits should both be set to 1. When SRQ is asserted, the I/O operation is aborted and the alarm program is scheduled as before, but control at this point returns to the user program with the error status set to 3.

The accompanying flow chart should answer any 'what if' questions regarding various combinations of S-R-D-E bits, buffered or unbuffered operation, as well as the effects of an EQT time out. It is assumed that the program 'ALARM' is linked to SRQ, and the user program has just made a write request to an HP-IB device. If the bus is buffered, control returns immediately to the user program and the error status is set to 0.


BEGIN
START I/O TO DEVICE
SRO ASSERTED ?
Y
N
S=1 ?
Y
N
ABORT I/O
SCHEDULE ALARM
R=1 ?
Y
N
BUS BUFFERED ?
Y
N
E=1 ?
Y
N
ERROR STATUS =3
"I/O NR LUxx"
OPER. UP, EQT ?
Y
N
I/O DONE ?
N
Y
WAS SRO ASSERTED ?
N
Y
SCHEDULE ALARM
ERROR STATUS =0
RETURN TO USER PROGRAM
EOT TIME-OUT ?
N
Y
ERROR STATUS =1
BUS BUFFERED ?
Y
N
E=1 ?
Y
N
"I/O NR LUxx"
RESET TIME-OUT CLOCK
OPER. UP, EQT ?
Y
N
WAS SRQ ASSERTED ?
N
Y
D=1 ?
Y
N
2ND EQT TIME-OUT ONLY ?
N
Y
"ILL INT xx"

HP-IB SPARKS PRODUCTIVITY IN MANUFACTURING APPLICATION

Larry Marafka
Westinghouse Electric
Engineering Computer Services

Hewlett-Packard's interface bus, commonly referred to as HP-IB, is an extremely important and useful tool in real time data acquisition.

The HP-IB can be employed to increase productivity with subsequent gains in profitability in many applications. This was an aim of Westinghouse when a Hewlett-Packard 1000 System with associated hardware was purchased to do computer aided rotor processing in a real time production environment.

Typically the rotors are approximately 3 to 6 feet in diameter, 25 to 45 feet in length, and weigh 50 to 200 tons. Prior to the development of an automated system for balancing the rotors, all data and calculations were processed manually. The project began by taking an in-depth look at two major areas of concern, namely investment and benefits.

Initially, management had to analyze the following items in terms of investment:

- Manpower requirements to implement the project.
- Adequate computer mainframe and other hardware required to accomplish the task.
- Real time software to support the computer and other hardware.
- Development costs associated with systems analysis and programming to automate the balancing procedure.
- A training mission to ensure balancers could use the system effectively, and communicate with other using departments.
- A commitment from line supervisors to enhance system development and follow specific disciplines outlined in an automated balance procedure.
- Establishment of project control checkpoints to review progress and discuss problems to minimize potential delays in implementing the project.

Our analysis of benefits to be realized from an automated balancing system encompassed these major items:

- Establishment of a standard method of balancing rotors using automated techniques via the HP-IB and real time processing.
- Improvement in data integrity and interpretation of the data in the decision making process.
- Work in progress inventory reductions through improved control and faster movement of material through the process.
- Better customer delivery by reducing the overall time to balance a rotor.
- Improvements in storing and retrieving data on rotors for reports required within our organization.
- Increasing our capability to provide customers with the information required by them concerning our ability to meet their specifications.

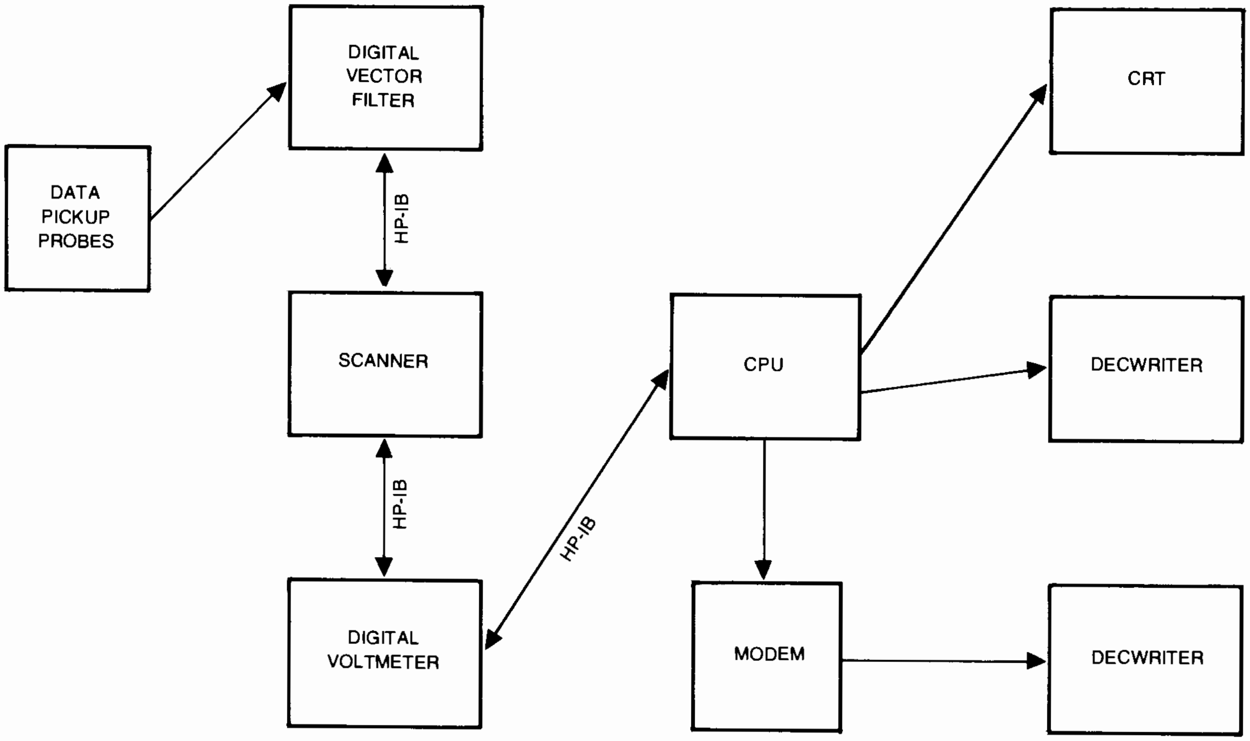
Management made a decision to purchase a Hewlett-Packard mini-computer system based on the following factors:

- Return on investment based on the analysis of costs versus benefits.
- Prior experience with Hewlett-Packard equipment used for data acquisition.
- Personnel already trained with knowledge of HP's hardware and software.
- Hewlett-Packard's ability to provide the Real Time Executive (RTE II) operating system.
- Hewlett-Packard's HP-IB capability including hardware and software.

From a systems viewpoint, consider the hardware and software required to accomplish automated balancing. The hardware varies with each application, but the concepts implied with HP-IB usage remain constant. That's really the major aim of this article, to provide an insight into some capabilities of the HP-IB that were used in the rotor balancing application.

Examine a system overview of the physical hardware:

HP 1000 System



The balancer is able to control activities and monitor rotor vibration levels from the cathode ray tube (CRT), which is a Hewlett-Packard 2645A with cassette cartridge units. This is the main communication link within the system. The operator enters data, initializes or aborts programs, responds to program questions, and monitors activities via this device. All data is captured on disc. This data is used in making calculations required to balance the rotor. The data is also stored for any future reference that may be necessary.

Hard copy output of data and required reports are printed on the decwriter connected to the central processing unit with RS 232 compatibility. Another decwriter is connected to the central processing unit via modem with automatic answer capability. This decwriter resides in an office remote from the HP 1000 system. With the modem connection to the system, personnel are able to execute programs and do program development from their offices. With the RTE operating system, this can be done concurrently with balancing production runs.

We insure balance program execution by assigning higher priority to production programs so they execute first. RTE handles the program swapping to disc and subsequent execution of lower priority jobs such as application program development.

Additional hardware includes Hewlett-Packard's 3495A scanner, HP's 3455A Digital Voltmeter, and Bentley Nevada's Digital Vector Filter. These devices are all compatible with the IEEE Standard 488-1975 instrumentation interface. Any instruments that meet the standard will operate satisfactorily on the bus.

Since this particular application involves a complex process, many programs, program segments, and subroutines were developed to assist in the automated balancing procedure. To illustrate, and gain an appreciation and insight into the "nuts and bolts" of coding using the HP-IB and related instrumentation, consider some portions of specific code. One of the major components is a Digital Vector Filter manufactured by Bentley Nevada. It is used to measure RPM, phase angle and amplitude of vibration at various positions on the rotor. The following code shows how our application used this device in an HP-IB environment:

```
FTN4,L
      SUBROUTINE DVF3(ARRAY,ZRUN,IRPM)
      COMPLEX ARRAY (9)
      DIMENSION IBITS(32),DP(2),IPH(2),AMP(2),IRNG(2),ICT(5)
      DATA ICMA/2HP#/
      DATA ITLK/2HOC/
      DATA IUNL/2H_?/
      DATA IIN/10B/
      DATA ICT/1403B,6014B,30060B,140300B,000000B/
C     ********************************************
C     *   SETTING VALUES IN 'IBITS' ARRAY TO     *
C     *   PROGRAM DVF II                         *
C     ********************************************
      DO 98 I=1,32
      IBITS(I)=2H
98    CONTINUE
      DO 99 I=1,8
      ARRAY(I)=CMPLX(0.0,0.0)
99    CONTINUE
      IBITS(1)=2H@J
      IBITS(17)=2HH@
      IBITS(18)=2H0U
      IBITS(19)=2H\@
      IBITS (26)=2HV2
      IRPM=0
C     *********************************************
C     *   CLEARING THE IB - ISSUING THE 'IFC'     *
C     *********************************************
      CALL EXEC(3,0000B+8)
C     *********************************************
C     *   ISSUING THE 'REMOTE ENABLE' COMMAND     *
C     *********************************************
      CALL EXEC(3,1600B+8)
```

```
C     ******************************
C     *  COMMANDING CPU TO TALK    *
C     *  COMMANDING DVF TO LISTEN  *
C     ******************************
      CALL EXEC(2,12000B+8,0,0,ICMA,-2)
C     *****************************************************
C     *  ISSUING PROGRAM COMMANDS TO DVF IN ASCII         *
C     *****************************************************
      CALL EXEC(2,12000B+8,IBITS,-64,0,0)
      CALL RELRG(0)
      CALL RELRG(ICT(1))
      CALL EXEC(12,0,2,0,-15)
      DO 76 IK=2,5
C     ********************
C     *  SENDING 'GET'   *
C     ********************
      CALL EXEC(2,12000B+8,0,0,IIN,-2)
C     *****************************************************
C     *   SENDING UNLISTEN AND UNTALK COMMANDS            *
C     *****************************************************
      CALL EXEC(2,12000B+8,0,0,IUNL,-2)
      CALL EXEC(12,0,1,0,-150)
C     *************************
C     *  ENABLE SERIAL POLL   *
C     *************************
      ISPE=30B
      CALL EXEC(2,12000B+8,0,0,ISPE,-2)
C     *****************************************************
C     *COMMAND DVF TO TALK AND CPU TO LISTEN              *
C     *****************************************************
      CALL EXEC(2,12000B+8,0,0,ITLK,-2)
C     *********************************
C     *  READ STATUS WORD ON BUS      *
C     *********************************
761   CALL EXEC (12,0,1,0,-40)
      CALL EXEC (1,12000B+8,ISTS,-2,0,0)
      IXIN = IAND(ISTS,2B)
      IF(IXIN .NE. 2B) GO TO 761
C     *************************
C     *  DISABLE SERIAL POLL  *
C     *************************
      ISPD=31B
      CALL EXEC(2,12000B+8,0,0,ISPD,-2)
C     *********************************************
C     *   COMMAND DVF TO TALK AND CPU TO LISTEN   *
C     *********************************************
      CALL EXEC(2,12000B+8,0,0,ITLK,-2)
C     ************************************
C     *  READ BINARY DATA FROM THE DVF   *
C     ************************************
      CALL EXEC(1,10100B+8,IBITS,32,0,0)
C     *****************************************
C     *  SENDING UNLISTEN AND UNTALK COMMANDS *
C     *****************************************
      CALL EXEC(2,12000B+8,0,0,IUNL,-2)
      CALL RELRG(ICT(IK))
      IF (IK .EQ. 5) GO TO 762
      CALL EXEC(12,0,2,0,-10)

            .
            .
            .
            .
```

The remaining part of the subroutine above is omitted since it simply converts the data into phase angle, RPM, and amplitude of vibration. The commented statements in the code explain exactly what's happening between the controller, the HP-IB, and the digital vector filter. Before attempting a program it is essential that the input/output characteristics of the devices on the bus be known. As an example, refer to the section of code labeled "READ BINARY DATA FROM THE DVF" which stores data into an array called IBITS. The control word that describes the device and gives information about the HP-IB driver is the second parameter listed, which is specified in octal form. All of the commands for using DVR37 are described in the HP-IB user's manual. The programming example just described is by no means the only possible coding scheme, it is merely presented to show one application of an HP-IB. It also verifies that all devices which are IEEE-488 bus compatible will work regardless of the manufacturer.

Note also the mode of programming used in this subroutine, namely direct I/O. In this programming mode, the programmer is responsible for all bus protocol. The programmer controls the devices and system controller by issuing 'TALK', 'LISTEN', 'READ', and 'WRITE' commands. Refer to the coding to show how a programmer controls the bus and devices on it.

Since we're operating in RTE-II with 32K words of memory, much of the coding is done using segments. Subsequently, a main program calls whatever segment is required to accomplish a portion of the balancing procedure. Some of the major tasks accomplished by individual segments include:

- Monitoring vibration levels of the rotor at all speeds.
- Checking temperatures and controlling a seasoning cycle where the rotor is heated for a specified time.
- Obtaining data and performing calculations on electrical characteristics of the rotor.
- Monitoring vibration levels during overspeed operations.
- Calculating and predicting balance moves based on the data gathered at various speeds.
- Monitoring transducer pressures and various temperatures.

It should be quite clear that a mini-computer system with supporting instrumentation interfaced with the HP-IB is indeed a powerful tool. This installation has only "scratched the surface" in employing the computer and related instrumentation in our shop environment to make gains in productivity. We have identified and are working in other areas where real time savings are obtainable. Considerable effort must be devoted to system design, problem definition, and programming to achieve the desired results. Application program development can be quite extensive depending on the requirements. On the project being described, program development costs have exceeded the hardware costs. This is not surprising in today's systems environment where the cost of hardware is decreasing, but software development and services increasing due to labor costs.

Practically all of our shop testing on production jobs is done manually. To expedite testing and processing data, the capability to do remote processing via the HP-IB emerged as a logical extension to the system described above. Once the original system was installed and running successfully, the requirements for remote processing from other shop locations were discussed with local HP sales personnel.

Additional hardware to provide a portable remote data acquisition testing capability was purchased. The equipment was rack mounted on a movable cart that could go to various shop locations, connected to proper electrical wiring, and data could then be processed using the Hewlett-Packard 3070A remote HP-IB capability. The new hardware that was required included an HP 2645A CRT with cassettes, an HP 3455A Digital Voltmeter, and an HP 3495 Scanner.

A new system generation and various application programs were written to accomplish defined tasks. Once completed, the remote HP-IB capability became an integral part of our expanded system. It is now possible to run several production jobs at the same time. RTE handles the program swapping to disc and other housekeeping type details. One of the reasons for mentioning the addition of new hardware is to highlight the ease of expanding an HP 1000 system and its HP-IB capability.

To summarize, below are listed the goals of this article:

- To provide the user with an insight onto an HP-IB application.
- To provide an overview of how we applied some HP-IB concepts.
- To provide some guidelines on system development using an HP-IB.
- To show one way that the HP-IB can be used remotely.
- To provide the potential user with some basic information on the HP-IB that can be a stepping stone for further investigation and use in designing a system.

In conclusion, the results obtained by investing the time and effort to employ the HP-IB can be very cost effective if an application requires instrumentation that may be interfaced to a computer with the HP-IB. Hopefully this article will serve as an incentive to potential users of the HP-IB to take action and reap the rewards of increased productivity.

RTE-IVB UPGRADE COURSE

Starting in September an RTE-IVB Upgrade course (22997A) will be offered at most domestic training centers and implemented shortly thereafter in Europe. This two day course is meant for those HP 1000 customers presently using RTE-IVA systems and thoroughly familiar with its operation. Major emphasis will be on describing the new features of RTE-IVB and Session Monitor. Material covered will serve the dual objective of teaching the user all about the new features, as well as instruct the system manager on how to install and maintain an RTE-IVB/Session Monitor system. The cost is $325. Contact your local sales office for registration information.

MORE NEW 1000 TRAINING CLASSES

September will also see the changeover from the old Disc-Based Operating System Course (22991A) to the new Session Monitor User's Course (22994A) as the mainstay of the HP 1000 technical training program. The previous two week format has, in fact, been expanded to three weeks with the addition of a separate one week System Manager Course (22995A). These new courses will be presented in proximity to each other, if not immediately following, to allow continuous training for individuals wishing to complete both classes during the same visit. Scheduling is always a matter of customer convenience, and any HP 1000 training class may be taken at any time as long as the student meets class prerequisites. Information on all customer courses, including dates, locations, descriptions and prerequisites can be found in the North American Customer Training Schedule brochure 5953-0841.

INTERNATIONAL TRAINING CENTER ADDRESSES

AUSTRIA
(Vienna)
Handelskai 52
Postfach 7
A 1205 Wien
Tel: (0222) 35 16 21-32
Telex: 75923
Cable: Hewpack Wien

AUSTRALIA
(Blackburn) B
CUSTOMER TRAINING CENTER
31-41 Joseph Street
Blackburn, Victoria, Australia
(Pymble) P
CUSTOMER TRAINING CENTER
31 Bridge Street
Pymble, New South Wales, Australia

BELGIUM
(Brussels)
Avenue du Col Vert, 1
Groenkraaglaan
B-1170
Brussels, Belgium
Tel: (02) 672 22 40

ENGLAND
(Altrincham) A
Navigation Road
Altrincham
Cheshire WA14 1NU
(Winnersh) W
King Street Lane
Winnersh, Workingham
Berkshire RG11 5 AR
Tel: Workingham 784774
Cable: Hewpie London
Telex: 8471789

FINLAND
(Helsinki)
Nahkahousuntie 5
00211 Helsinki 21
Tel: 90-692 30 31

FRANCE
(Grenoble) G
5, avenue Raymond-Chanas
38320 Eybens
Tel: (76) 25-81-41
Telex: 980124
(Orsay) O
Quartier de Courtaboeuf
Boite Postale No. 6
F-91401-Orsay
Tel: (01) 907 7825

GERMANY
(Boeblingen)
Kundenschulung
Herrenbergerstrasse 110
D-7030 Boeblingen, Wurttemberg
Tel: (07031) 667-1
Telex: 07265739
Cable: HEPAG

ITALY
(Milan)
Via Amerigo Vespucci, 2
20124 Milan
Tel: (2) 62 51
Cable: HEWPACKIT Milano
Telex: 32046

JAPAN
(Osaka) O
Chuo Building
5-4-20 Nishinakajima
Yodogawa-Ku, Osaki-shi
Osaka, 532 Japan
Tel: 06-304-6021
Telex: 523-3624 YHP OSA
(Tokyo) T
2205 Takaido Higashi 3-chome
Suginami-Ku, Tokyo 168
Tel: 03-33-8111
Telex: 232-2024 YHP MARKET TOK

NETHERLANDS
(Amsterdam)
Van Heuven Goedhartlaan 121
Amstelveen 1134
Netherlands
Tel: 02 672 22 40

SPAIN
(Madrid)
Jerez No. 3
E-Madrid 16
Tel: (1) 458 26 00
Telex: 23515 hpe

SWEDEN
(Stockholm)
Enighetsvagen 1-3, Fack
S-161 20 Bromma 20
Tel: (08) 730 05 50
Cable: MEASUREMENTS
Telex: 10721

SWITZERLAND
(Zurich)
19 Chemin Chateau Bloc
1219 Le Lignon Geneve
Tel: 022/96 03 22

For course prerequisites and registration information contact one of the HP training centers listed above.

INTERNATIONAL CUSTOMER TRAINING SCHEDULES 79/80

	AUSTRIA	AUSTRALIA	BELGIUM	ENGLAND	FINLAND	FRANCE	GERMANY	ITALY	JAPAN	NETHERLANDS	SPAIN	SWEDEN	SWITZERLAND
22941A 21MX/XE Maint. 5 days/$500						Oct 22 (G)							
22943A 7970B/E Maint. 5 days/$500													
22945A 7905/06 Maint. 5 days/$500						Oct 29 (G)							
22951B Intro to HP 1000 4 days/$400	Sep 03			Sept 19 (W)			Sept 24	Sept 10 (R) Oct 22 (M)		Aug 27 Nov 19 Feb 11 May 05		Oct 08	
22951B-H01 FORTRAN IV 5 days							Oct 08				Oct 15		
22952B 1000 ASMB 5 days/$500	Sep 24	Oct 08 (P) Nov 12 (B)	Sept 10	Oct 29 (W)	Nov 11	Oct 08 (O)	Sep 03	Oct 15 (R) Dec 17 (M)	Jun 11 (T)	Dec 17 Apr 21	Oct 22	Sep 10 Nov 12	
22961B DS/1000 Theory of Operation 4 days/$500							Oct 01			Jan 28			
22962B DS/1000 to HP 3000 Theory of Operation 1 day/$100							Oct 05			Sep 28			
22965B RTE-II/III 10 days/$1000					Sept 10 (O) Oct 15 (O) Dec 03 (O)						Oct 29		
22969A Distributed Systems 5 days/$500						Nov 26 (O)							
22977 IMAGE 5 days/$500		Aug 20 (B) Nov 12 (P) Dec 03 (B)		Oct 15 (A)	Dec 23	Sep 24 (O)				Oct 08 Mar 17 Jun 09	Nov 12		
22980C HP-IB Interface With HP 1000 4 days/$400										Aug 20 Feb 04			
22983B 21MX/E Microprogramming 5 days/$500						Oct 01 (O)							
22984A 7920 Maint. 5 days/$5000													
22985A RTE-M 5 days/$500							Sep 10						
22987A DS/1000 User's Course 5 days/$500		Oct 22 (P)		Oct 22 (W)			Sep 17			Nov 05 Jun 23			
22990A RTE Driver Writing 3 days/$300		Aug 06 (B) Oct 01 (P) Nov 19 (B)								Oct 01 Dec 10 Apr 28			

▨ Mature Product Courses

INTERNATIONAL CUSTOMER TRAINING SCHEDULES 79/80 Continued

	AUSTRIA	AUSTRALIA	BELGIUM	ENGLAND	FINLAND	FRANCE	GERMANY	ITALY	JAPAN	NETHERLANDS	SPAIN	SWEDEN	SWITZERLAND
22991A RTE-IVA 10 days/$1000	Sep 10	Sep 17 (P) Oct 22 (B)	Oct 01				Aug 20 Sept 10 Oct 01 Oct 15	Sept 24 (R) Nov 12 (M)		Sept 03 Oct 15 Nov 26 Jan 07 Feb 25 Mar 31 May 19	Oct 29	Sept 10 Oct 15 Nov 19	Sept 10
22992A HP 1000 Memory RTE 10 days/$1000							Sept 10						
22993A IMAGE 5 days/$500													
22994A Session Monitor User 10 days/$1000					Sept 23 Nov 25								
22995A System Manager 5 days/$500													
22996A RTE-IVA-IVB Upgrade 2 days/$325													
22997A Advanced RTE 5 days/$800													
40270A Intro to HP Computers 5 days						Nov 05 (O)							
91302A 2645 Maint. 3 days/$300													

▨ Mature Product Courses

NEW HP-IB APPLICATION NOTE SERIES 401 BRINGS IT TOGETHER!

Neal Kuhn/Applications Development

At last! The HP-IB support that you have been asking for: comprehensive instructions how to integrate and program some of our most popular HP-IB instruments with an HP 1000 computer. This series of application notes is aimed at users who have some knowledge of the computer but are not too familiar with automating instruments.

Following a logical procedure, these instructions take the user step-by-step from setting addresses, through setup and on-line programming, to sophisticated performance comparisons. *All program listings are included* — programs that have been developed and tested by Data Systems' Applications Group. They can save you hours of software development.

AN 401-1, HP 1000/HP-IB PROGRAMMING PROCEDURES

AN 401-1 (5953-2800) is the main overview note and should be read first. It supplies the general prerequisite information and software utilities which are implemented in the rest of the series.

AN 401-1 presents an easy-to-follow outline that simplifies HP-IB instrument operation with the HP 1000. It explains what hardware to use, how to set the device address, how to configure your system (software requirements), and even gives performance graphs showing how many readings per second an instrument can take.

The actual programming examples range from the most fundamental and interactive, to high performance routines that will take your breath away. If the instrument is somewhat older and does not always behave as expected, these peculiarities are documented and listings for workaround solutions are given.

AN 401-1 answers commonly asked questions such as:

"How do I determine LU assignment?"
"What about buffering and should I use it?"
"How do I set limits on time-out and how do I use it?"
"What does the Device Configuration Word look like, and how do I set it?"
"Does the device have service request capability and how can I optimize its use?"
"Should I use the interrupt technique or DMA?"

For those of you who have been asking us to make the computer more friendly, check the following new sections:

- **How to send commands to the instrument and take readings** just as with a desktop computer. We show how to use **File Manager** commands and the return key. Programming, compilation, and relocation are not required for checking out your instrument.

- **A new HP-IB Status/Configuration Utility** is given (get it from the contributed library). It describes EQT's and LU's and tells the user how everything is set up on the bus. This greatly simplifies the operation of multiple devices on the bus.

- A program is given (from the contributed library) that **verifies device addresses and LU assignments.** Even if you don't know what address is set on your device, just plug it on the bus and the computer will tell you its address and automatically assign it an LU number. What could be more friendly?

NOT JUST ONE AP NOTE, BUT 21 OF THEM

AN 401-1 is supplemented by detailed instrument-specific programming guides. These are separate modules so that you can select those pertinent to your individual needs. Contact your HP representative for a copy.

Application Note	Content	Document Number
401-1	HP 1000/HP-IB Programming Procedures	5953-2800
401-2	59307 VHF Switch/HP 1000 Computer	5953-2801
401-3	5345A Counter/HP 1000 Computer	5953-2802
401-4	5342A Microwave Counter/HP 1000 Computer	5953-2803
401-5	5328A Counter/HP 1000 Computer	5953-2804
401-6	3438A Digital Multimeter/HP 1000 Computer	5953-2805
401-7	3455A Digital Multimeter/HP 1000 Computer	5953-2806
401-8	59309A Digital Clock/HP 1000 Computer	5953-2807
401-9	6002A Power Supply/HP 1000 Computer	5953-2808
401-10	3437A Digital Voltmeter/HP 1000 Computer	5953-2809
401-11	3495A Scanner/HP 1000 Computer	5953-2810
401-12	3582A Spectrum Analyzer/HP 1000 Computer	5953-2811
401-13	3325A Function Generator/HP 1000 Computer	5953-2812
401-14	4262A Digital LCR Meter/HP 1000 Computer	5953-2813
401-15	8672A Synthesized Signal Generator/HP 1000 Computer	5953-2814
401-16	436A Microwave Power Meter/HP 1000 Computer	5953-2815
401-17	8620C Sweep Oscillator/HP 1000 Computer	5953-2816
401-18	59306A Relay Actuator/HP 1000 Computer	5953-2817
401-19	8660C Synthesized Signal Generator	5953-2818
401-20	9871A Character Impact Printer	5953-2819
401-21	6942A Multiprogrammer II	5953-2820

RTE-IVB/SESSION MONITOR — A TRUE MULTI-USER OPERATING SYSTEM

John Koskinen/HP Data Systems Division

Data Systems Division has recently introduced an enhanced, version of the RTE operating system. This enhancement helps increase programmer and general user productivity in a multi-user work environment. The enhanced operating system includes such features as time slicing, user and group security for session devices and files, and session accessible spooling. The system manager now has a very useful tool to properly allocate and protect resources for multi-function and multi-application work stations.

COMPUTER WORK STATION

RTE has grown in use and power over the years from a dedicated single user/single application system to a multi-user/multi-application system. With RTE-IVB/Session Monitor, the use of RTE in a disperse and varied work environment has been made significantly easier. Session Monitor allows the system manager to construct and configure computer work stations tailored for specific users and specific applications. These work station configurations are invoked at the time a user logs on to the system with appropriate identification (and passwords if needed). The computer work station provides a mechanism for both allocating and protecting system resources such as disc cartridges, Distributed System/1000 lines, line printers, graphics devices, and on-line instrumentation. The basic elements a system manager uses to configure a work station are: user capability levels, session device tables, and user Hello files.

CAPABILITY LEVELS

There are sixty levels of capability defined roughly into five groups. Each group in ascending order is given an increasing amount of access to FMGR and system commands. The lowest level of capability only allows a user to log on to the system, and transfer of control must go to a procedure or transfer file. The highest level allows access to the entire system.

SESSION DEVICE TABLES

Associated with each computer work station is a set of devices which are defined by the system manager. The set of devices may be configured into the work station as a function of the physical location of the log-on terminal and/or as a function of the user or group specified at log-on time. These devices may be sharable system resources like disc drives, line printers, and magnetic tape units. The devices may also be exclusive devices such as on-line instrumentation used for dedicated applications.

A very important feature of the session device table is that the device logical unit numbers can always be the same for a user wherever, and whenever he logs on to the system. Session Monitor software maps session local logical unit numbers to system logical and physical unit numbers. The benefit here is that programs can now be written without any special code needed to do device logical unit housekeeping.

The user's program can be written with common or standard logical unit numbers: device housekeeping is done by the session device table.

This leads to some excellent application benefits:

1. Many work stations may run the same application — The code developed for these work stations can be put together independent of physical device locations. The actual physical devices are mapped into the session where the user logs on.

2. One work station may support many different applications — As different users log on to the system, they have different I/O device requirements. At log-on time, each user, via the session device table, accesses only the devices needed, and other devices are secured from accidental access or tampering.

USER HELLO FILE

The Session Monitor allows the system manager or user to create a procedure file that is executed immedialely at log-on time. This file is the user's own Hello File. It can be used to initialize the work station for a particular job: loading the 2645 soft keys, setting up spool logical units, or running an initialization program. At a completely dedicated work station the Hello file can start a turnkey program so that the user doesn't have to be trained to use FMGR or system commands.

PROGRAM TIME SLICING

One of the resources to be shared in a multi-user system is, of course, CPU time. RTE-IVB has implemented a time slice capability that allows the system manager to load balance the system. The scheduling and dispatching of programs is still essentially done on a priority basis. Foreground programs will be time sliced. Background programs will be time sliced along with any other programs running at the same priority. Programs at low priorities recieve CPU time only when all higher priority programs have been satisfied. The point here is that the system manager knows more about the job mix than can be assumed by the RTE scheduler, and thus can place priorities on programs so as to optimize total machine throughput for his system.

SESSION SPOOLING

Another feature that RTE-IVB provides for multiple users is session spooling of unit record devices like the line printer, card readers, and the tape drive. A user, under session control, simply specifies that certain list or input devices are to be spooled. The spooler automatically sets up disc spool files and, in the case of the line printer, prints the files at log-off time or when the spool LU is released.

IN SUMMARY

All of the new features introduced with RTE-IVB/Session Monitor provide a new dimension and new direction for HP 1000 systems. While the systems are still excellent choices for dedicated process control and similar applications, the HP 1000 can now be used for a much wider set of applications; applications that demand a controlled and coordinated access to system resources from many diverse users.

WATCH FOR NEWS OF AN INTERNATIONAL HP 1000 USER GROUP

A lot of interest has been expressed recently regarding the establishment of an International HP 1000 User Group. Such a group would benefit users worldwide by centralizing activities such as the Contributed Library, a newsletter, a technical journal, and an international convention once a year.

A small group of users from around the world will be getting together in Cupertino on August 23 and 24 to try and start such an international organization. Thus, you should be hearing more about this very exciting possibility in the very near future. In the meantime, if you're bursting with ideas on this subject, or you'd like to get involved in setting up the group, send a letter to:

Editor, Communicator 1000
Data Systems Division
Hewlett-Packard Co.
11000 Wolfe Rd.
Cupertino, CA 95014
U. S. A.

JOIN AN HP 1000 USER GROUP!

Ever wonder how other HP users have used the HP 1000 in application areas similar to your own? Have a special program or driver you'd like to share with other users? Interested in hearing about new developments in HP 1000 hardware and software?

If your answer to any one of these questions is YES, then an HP 1000 user group might be just the thing for you. These and other similar activities are carried on regularly as part of the function of the many HP 1000 user groups that exist around the world. There's a good chance that there's a user group right near your location! To get in on the action, just check the list below, and contact the group nearest you to find out when and where the next meeting will be held.

Or, if there isn't a group near you, why not start one? The Communicator/1000 can help you out by announcing the creation of new groups. Just send a letter — c/o Editor, HP 1000 Communicator, — with the name of your new group and the means by which other users can join. We'll add your group to our list and publish it in the next issue of the Communicator.

Here are the groups that we know of as of August, 1979. (If your group is missing, send the Communicator/1000 editor all of the appropriate information, and we'll update our list.)

NORTH AMERICAN HP 1000 USER GROUPS

		Next Meeting	
Area	**User Group Contact**	**Date**	**Location**
Boston	LEXUS P.O. Box 1000 Norwood, Mass. 02062	Not selected yet (see article on next page about RTE users workshop at URI on October 11, 12)	
New York/New Jersey	Paul Miller Corp. Computer Systems 675 Line Road Aberdeen, N.J. 07746 (201) 583-4422	Not selected yet	

Philadelphia	Dr. Barry Perlman RCA Laboratories P.O. Box 432 Princeton, N.J. 08540	Sept 17.	Bell Laboratories Allentown, PA
Pittsburgh	Eric Belmont Alliance Research Ctr. 1562 Beeson St. Alliance, Ohio 44601 (216) 821-9110 X417	Sept. 5 Nov. 7	Holiday Inn Pittsburgh —
San Diego	Jim Metts Hewlett-Packard Co. P.O. Box 23333 San Diego, CA 92123	Not selected yet	
Washington/Baltimore	Paul Toltavull Hewlett-Packard Co. 2 Choke Cherry Rd. Rockville, MD. 20850	Sept. 27	Hewlett-Packard Co. 7121 Standard Dr. Hanover, Maryland
General Electric Co. (GE employees only)	Stu Troupe Special Purpose Computer Ctr. General Electric Co. 1285 Boston Ave. Bridgeport, Conn. 06602	Oct. 1-3	Hewlett-Packard Co. 5201 Tollview Ave. Rolling Meadows, IL.

EUROPEAN HP 1000 USER GROUPS

London	Rob Porter Hewlett-Packard Ltd. King Street Lane Winnersh, Wokingham Berkshire, RG11 5AR England (734) 784 774	Not selected yet
Amsterdam	Mr. Van Puten Institute of Public Health Bilthoven Anthony Van Leeuwenhoeklaan 9 The Netherlands	Not selected yet

RTE USERS WORKSHOP AT URI — OCTOBER 11, 12

There has recently been interest in holding a second RTE users workshop in the Northeastern U.S./Eastern Canada area. The first workshop, organized by Ron McNab, was held at the Bedford Institute of Oceanography in Halifax, Nova Scotia in December of 1977. The second workshop will use a similar format and will focus on gathering RTE users in order to provide an opportunity for exchange of information and experiences.

The workshop is scheduled for October 11, 1979 at the Graduate School of Oceanography of the University of Rhode Island. The conference facilities of the university will be available to us for the workshop. We plan to have 15 minute informal presentations followed by 5 minute discussion periods in the mornings. Afternoons will be left open for hardware and software demonstrations, discussions, exchange of programs, etc.

Some topics which we feel may be of interest for presentations and discussion are:

Distributed Systems
Microcode
Graphics
Word Processing
Data Base Management
Microprocessor Emulation
RJE
Intelligent Terminals
IEEE or HP-IB Developments
Statistical Software
EMA

Any other topics related to HP RTE will, of course, be welcome. If you are interested in attending this workshop or in participating with a presentation, please contact:

Chris Polloni or Diane Dow
227 Watkins
Graduate School of Oceanography
Narragansett Bay Campus
University of Rhode Island
Kingston, R.I. 02881
(401) 792-6116

LAST MINUTE NOTE: Just before press time, we received an update on the status of the workshop. Currently, about 40 people have agreed to participate, with day 1 activities concentrating on Data Acquisition, and day 2 on Word Processing and Graphics.

Although every effort is made to ensure the accuracy of the data presented in the **Communicator,** Hewlett-Packard cannot assume liability for the information contained herein.

Prices quoted apply only in U.S.A. If outside the U.S., contact your local sales and service office for prices in your country.

Printed in U.S.A. 8/79

Part No. 5951-6111