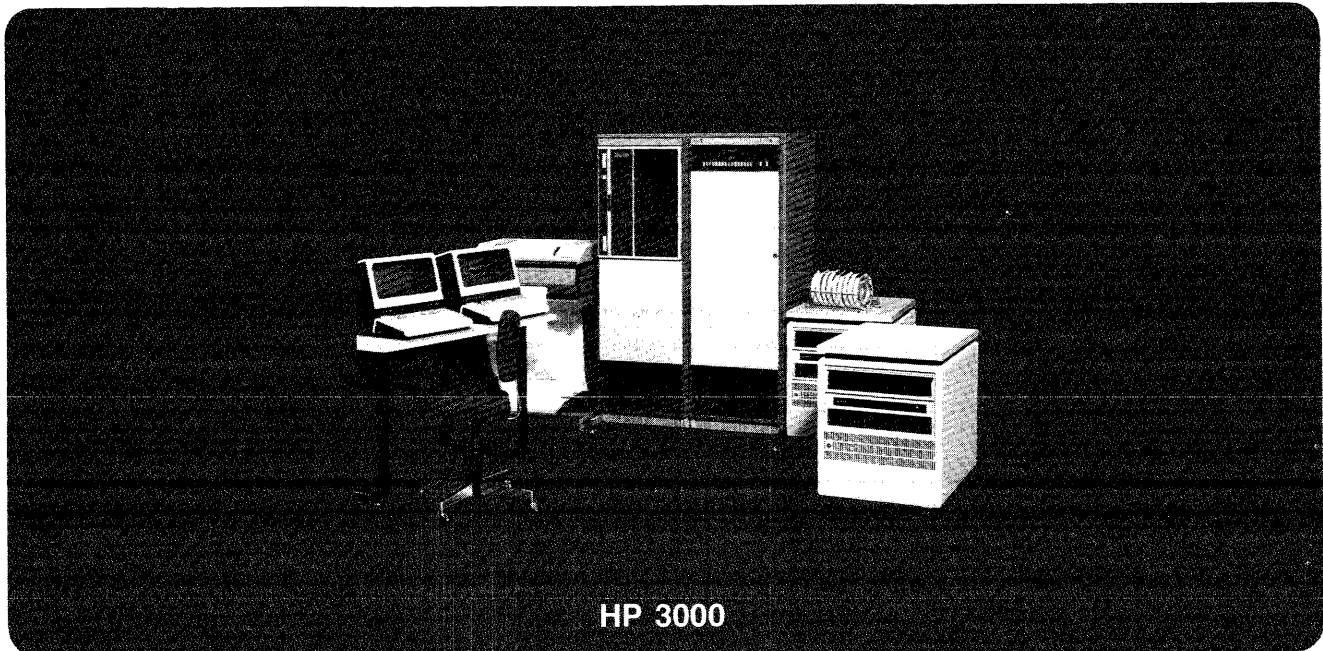# computer systems
COMMUNICATOR

HP 3000

HP 2000
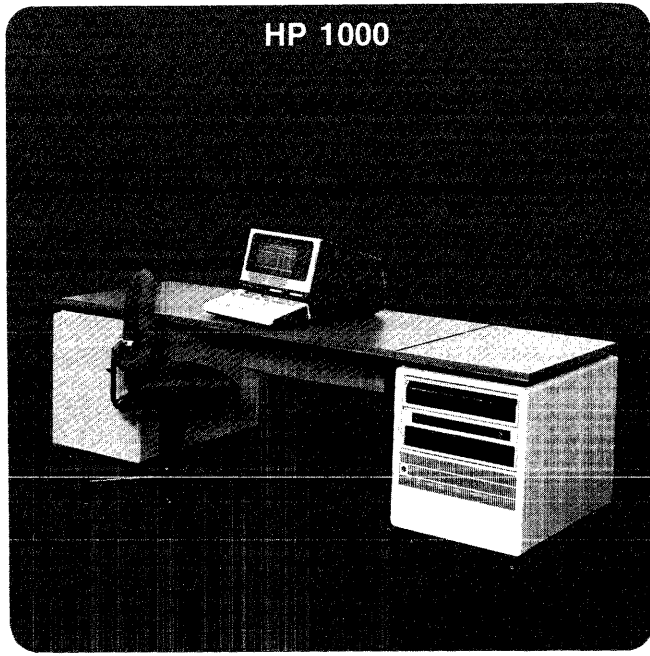
HP 1000

HP3000 Computer Systems

LANGUAGES

COBOL · BASIC · FORTRAN · APL · SPL · RPG

HP 3000

HP 1000   HP 2000

editor's note

The January-February issue of the Computer Systems **Communicator** features APL, A Programming Language. Introduction of APL/3000 on the HP 3000 Series II was announced at the International APL Conference in Ottawa, Canada, September, 1976. APL/3000 affords the attributes of APL on a low cost computer.
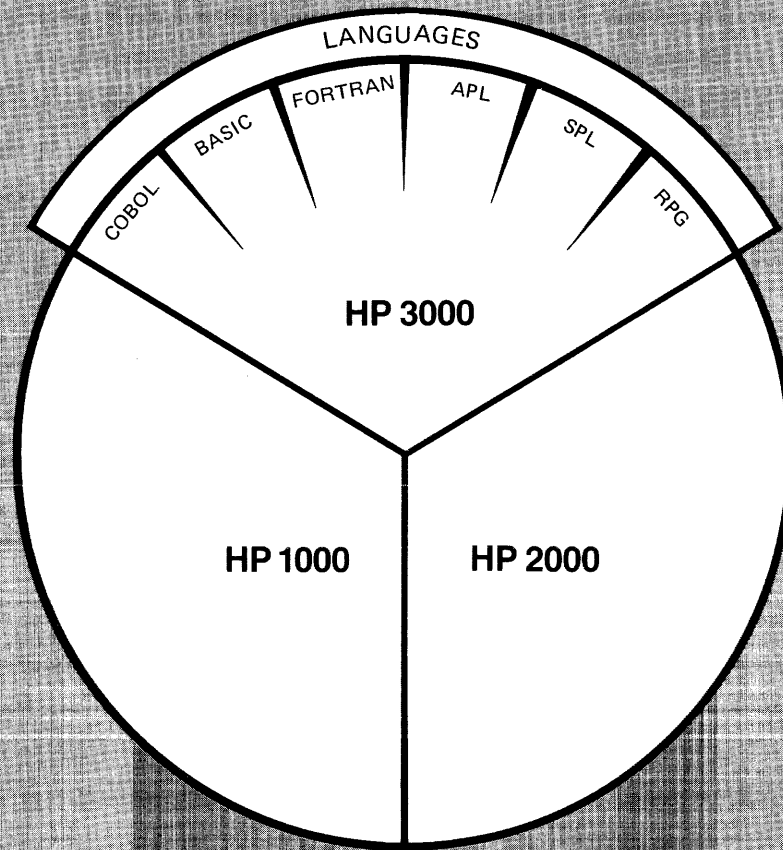
APL is a powerful and concise programming tool with applications in business, education and engineering, plus it provides an easy convenient solution to data processing problems. Terse syntax and dynamic data structures combine to facilitate programming, simplify debugging and enhance readability.

Highlighting the broad spectrum of other topics in this issue of the **Communicator,** articles investigate Micropro- gramming, 7905 Disc I/O Optimization, Line Printer Selection for HP 2000 Remote Job Entry, Segmentation in COBOL, and Calling SPL from RPG and calling COBOL from SPL.

Several new manuals and pocket guides are now available and have been briefly summarized in the Bulletins of the 1000, 2000, and 3000 Computer Systems sections. Be sure to look over these to keep your documentation up-to-date.

The 1000, 2000 and 3000 system training schedules have been updated for the U.S. and international locations. Classes at international locations have been listed into the summer months, where dates were available to give overseas installations scheduling information. We hope it proves useful to your planning.

Please contact the editor if you have new ideas or can suggest changes in the content or format of the **COMMUNICATOR** which will foster the creation of a quality publication that meets the informational needs of HP Computer Systems users.

Address your correspondence to:

Editor Janis Andrews
Computer Systems Communicator
HP General Systems Division
5303 Stevens Creek Blvd.
Santa Clara, CA 95050

You may also contact the appropriate divisional technical editors as follows:

- HP 1000 System (9600/9700)
  Gary Gubitz
  HP Data Systems Division
  11000 Wolfe Road
  Cupertino, CA 95014

- HP 2000 Computer System
  Dan Jorgenson
  HP General Systems Division
  5303 Stevens Creek Blvd.
  Santa Clara, CA 95050

- HP 3000 Computer System
  Janis Andrews
  HP General Systems Division
  5303 Stevens Creek Blvd.
  Santa Clara, CA 95050

contents ═══════════════════════════════════

software tips
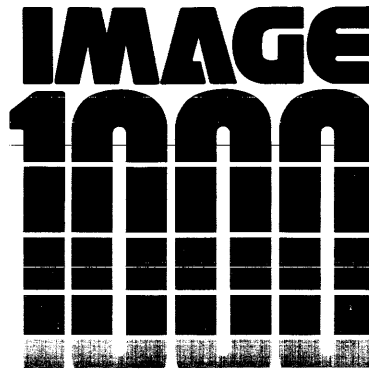
IMAGE 1000

about the HP 1000

IMAGE 1000

MULTIPROGRAM DELETIONS AND ADDITIONS OF RECORDS

Jim Schultz
HP Data Systems

It is now possible to add or delete date base records from several programs concurrently. Previously only one program could add or delete records; the program had to have opened the data base in mode 3, which is an exclusive open.

The design of this feature is based on the fact that information in the root file gets modified during the use of the data base management subroutine, and occurs while deleting and adding records. Moving this information to a central location, which is accessible by all programs, allows IMAGE to have a multiprogram record delete and add capability. This capability was accomplished by providing centralized storage of all information in the root file, that is modified by a 'DBPUT' or DBDEL'.

What happens when a user opens a data base in mode 2? First, the root file is read into the memory space immediately following the program. Second, all modifiable information in the root file is extracted and written out to RTE system available memory. Third, the table in the memory resident library that indicates all activity regarding mode 2 operations for the data base is set up. The activity table looks like this:

N	A	
M	E	Data Base Name
X	X	
Class number		Class number for information written to system available memory.
Resource number		Resource number to be set or cleared by lock & unlock.
Count		Count of programs opened in mode 2.

There are 4 entries in this table allowing four separate data bases to be open in mode 2 simultaneously.

When a program prepares to add or delete a record it must first lock the data base. This causes the information in system available memory to be extracted and stored in the programs root file in memory and also posts all data sets. The resource number is locked preventing any other user from locking the data base and adding or deleting records. When the program calls 'DBPUT' or 'DBDEL' a check is made to determine if the open mode was 2 or 3, if 2 a check is also made to see if the data base is locked. When the delete is completed the information is put in the root file that was changed, rewritten out to system available memory, and all data sets are posted. At this point, the data base in conjunction with the information in system available memory reflect all charges that were made while the data base was locked. Also, when the data base was locked any users that may have attempted to lock the data base (and were suspended) are rescheduled and allowed to perform their operations.

The activity table has a counter which keeps track of the number of programs that have the data base open in mode 2. The first user to open the data base in mode 2 sets up the activity table and the last one to close the data base clears it out releasing the class and resource numbers and obliterating the data base name. Any programs opening the data base in mode 2 after the activity table has been set-up merely increment the number of users.

Two new utilities have been added to IMAGE:

1. **DBSPA** — Indicates the number of records in a data set, number of records used and number of records available. It also shows any discrepancies which might indicate a bad data base. base.

2. **RECOV** — Indicates the number of programs that have the data base open and permits users who opened the data base in mode 2 to close it in the event of abnormal program termination.

about the HP 1000

HELPFUL HINTS:

1. Mode 2 usage of the data base has slightly more overhead when performing deletes and puts. This is because data has to be moved to and from system available memory, and must be posted from the data control blocks to the disc.

2. The utility RECOV should be run prior to shutting down the system to determine if any users terminated abnormally and did not close the data base, leaving information in system available memory. If the data base is still open RECOV will force the data base closed.

COMPUTERS
HP-IB
INSTRUMENTATION
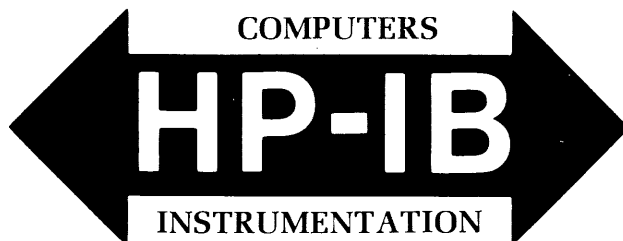
HP-IB TREKIE

BUS TOPOLOGY AND HANDSHAKE ANALYSIS
ARTICLE #2

Larry W. Smith
HP Data Systems

This is the second of a series of five articles relating to performance characteristics of the HP-IB. This article will deal with typical instrument worst-case configurations and an analysis on the handshake process. Recall, this article and the ones to follow outline efforts at Data Systems Division to arrive at system guidelines for optimizing HP-IB performance under worst case conditions. It also might be noted that results shown in this article were taken as the result of extensive computer simulation using ASTAP (Advanced Statistical Analysis Program). The system elements which were modeled are the driver, the receiver termination, and the cable. For more information relating to the model types used, please write a letter to the editor of the **Communicator**, HP 1000 Group.

BUS TOPOLOGY

The worst-case topology can be defined as the configuration of devices and cables which results in the longest settling time for signals on the bus. It is not straightforward to deduce this worst-case configuration since a topology with many cable branches would require several signal reflections (i.e. long "ringing" times) for the signal to stabilize, while a linear topology would maximize the propagation time between the end points of the cable. Extensive computer simulation of the various possibilities indicates that a linear connection scheme may be used as the worst-case. This is due to the somewhat surprising fact that the signal often settles last at a point near the driver.

It is convenient for worst-case analysis that a linear topology yields the longest settling time, since that configuration also maximizes the signal propagation delays. The

settling time affects the time a talker must allow for the data lines to stabilize before the data valid (DAV) message is asserted while the propagation delay affects the time required for the DAV signal to reach the listener. Since DAV is a negative-true signal and the line drivers have a small impedance in the low state, the signal must travel only one length of the cable for all devices to recognize it. Therefore, the worst-case propagation delay is 9.5 NSEC per meter of cable.

HANDSHAKE ANALYSIS

The figure below, HP-IB Data Handshakes, shows message timing during a data handshake. In this diagram, a complete cycle is defined as the time between successive assertions of the DAV message by the source. T1 is the time required for the source to recognize that the data has been accepted after DAV goes true. T1 consists of three components:

1. The time required for the acceptor to recognize DAV after that message has been sent by the source. This time is determined by the propagation delay of the cable and will be designated by TPD.

2. The device-dependent delay required for the device to accept the data and assert DAC after recognition of the DAV message. This time will be called TA1 (1st acceptor delay).

3. The time required for the open-collector line, which carries the DAC message to rise to a high voltage and be recognized by the source. This time is called TOC.

T2 is the device-dependent delay required for the source to output a new data byte after the DAC message has been received. At the end of T2 the source will be in the SDYS state of its SH state diagram.

T3 is the data settling time after which the source may assert DAV if the acceptor is ready. In order for the acceptor to appear ready, it must have sensed the removal of the previous DAV and sent the RFD message. The open-collector line carrying RFD must then have risen to a high voltage. If the acceptor appears ready before T3 expires, the handshake speed may be called "source-limited", since T3 is a delay at the source end of the bus. The total handshake time is then equal to

$$TSL = TPD + TA1 + TOC + T2 + T3$$

The standard directly specifies only T3, which currently must be 500 NSEC for tri-state drivers.

If the acceptor does not appear ready before T3 expires, the handshake may be termed "acceptor-limited". Even if the acceptor hardware is fast enough to be ready immediately after DAV is removed, an acceptor-limited cycle can occur due to the properties of the BUS. For example, an open-collector signal can require up to 814 NSEC rise time. In addition, the removel of the DAV message requires a low-to-high signal transition, and the time required for such a signal to reach its stabilized level at all points on the BUS can be equivalent to the data setting time, T3. Even if DAV settles rapidly, the signal must propagate down the total length of the BUS before a distant acceptor can return RFD. Therefore, if the worst-case DAV settling is ignored, acceptor-limited cycles will occur if

$$T3 < TPD + TOC$$

If the acceptor does not return RFD immediately when DAV is removed, that device-dependent delay (herein called TA2) must be added to the total handshake time in the acceptor-limited case. In the below figure, this time combines with TPD and TOC to form T5. T4 is the delay between DAC and the removal of DAV by the source. (Note that T4 and the TPD component of T5 may not contribute to the handshake time if TA2 is defined as the delay from the acceptor's assertion of DAC to the assertion of RFD. Such a definition would change the condition for acceptor-limited cycles accordingly.) The total acceptor-limited handshake time is

$$TAL = TPD + TA1 + TOC + T4 + TPD + TA2 + TOC$$

If all device-dependent delays (source and acceptor) are lumped into one term, the handshake times are

$$TSL = TPD + TOC + T3 + TDDS$$

$$TAL = 2TPD + 2TOC + TDDA$$

where TDDS are the device delays in the source-limited case and TDDA are the delays in the acceptor-limited case. Thus,

$$TPD = M \sqrt{0.6 \left(150 + 50 \frac{N}{M}\right)}$$

$$TOC = 116 + 349 \frac{M}{N}$$

where N is the number of devices evenly distributed over M meters of cable.

SPEED GUIDELINES

The results of this study may be used as guidelines for achieving a desired handshake speed as a function of cable length and number of devices. The following curves in figures 1 through 7 represent such guidelines for one megabyte operation with various device-dependent delays. That is, if TSI and TAL are 1000 NSEC, a curve of M versus N results from each value of TDD. Note that the definition of TDD depends upon whether the handshake is source-limited or acceptor-limited, even though both modes are plotted on the same graph. These curves assume that 350 NSEC is a valid settling time if the cable length is limited to one meter per device load. Worst-case conditions are used throughout. The regions of the curves are defined as follows:

Region 1: Illegal operation, since the total cable length exceeds two meters per device.

Region 2: 500 NSEC settling time if required. One megabyte operation cannot be achieved under worst-case conditions.

Region 3: 350 NSEC settling time if permitted. One megabyte operation cannot be achieved under worst-case conditions.

Region 4: One megabyte operation is possible under worst-case conditions if 350 NSEC settling time is used.

Region 5: One megabyte operation is possible using 500 NSEC settling time.

The curves of Figures 1 through 5 are combined into Figure 6. If the settling time is 350 NSEC, the handshake will be source-limited for device delays greater than 300 NSEC (assuming a one megabyte data rate.) This explains the slope discontinuities in Figure 6. If the settling time is 500 NSEC, the handshake is essentially always source-limited.

Note that Figure 6 was derived for all conditions being worst-case as permitted by the standard. The conditions may be redefined to represent typical values or perhaps worst-case values for the particular devices and cables being used. For example, Figure 7 results if the following "typical" conditions prevail:

1. The cable is described by the typical parameters given in the previous article.

2. The device capacitance is still the worst-case of 50 PF. The cable propagation for a linear topology of M meters with N evenly distributed devices is then

$$TPD = M \sqrt{0.482 \left(131 + 50 \, \frac{N}{M}\right)}$$

3. The devices' resistive termination is midway between the worst-case and best-case allowed by the standard and is defined by the straight line through the points

    0.4 V, 2.25 MA
    3.1 V, 0 MA

The resulting circuit is

5v

1.94K

3.16K
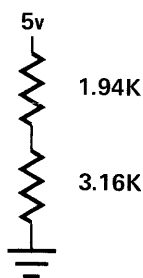
4. The high-level threshold for the BUS receivers is 1.7 volts, and the saturation voltage of the tri-state drivers is 0.4 volts. The open collector rise time is THUR

$$TOC = 39.4 + 103 \, \frac{M}{N}$$

CONCLUSION

The study has indicated that the HP-IB handshake speed is dramatically affected when worst-case conditions prevail. In order to allow a designer to optimize the BUS data rate under such conditions, the following recommendations seem reasonable:

1. Change the standard to permit the use of Schottky devices by allowing the driver output voltage to be 0.5 volts at 48 MA sink current.

2. Change the standard to allow a maximum device capacitance of 50 PF.

3. Change the standard to permit a data settling time of 350 NSEC if tri-state drivers are used and the total cable length is limited to one meter per equivalent device load.

4. Change the wording of the standard to allow a BUS attachment to present a DC load equivalent to multiple devices if it is known that the total number of attachments will be limited. This will permit a controller to simulate several best-case device loads. This best-case resistive load imparts the advantages of extra device attachments without introducing device capacitance which affects open collector rise times and propagation delays.

5. Include some speed guidelines in the standard. These guidelines could perhaps include a set of curves showing the expected conditions rather than worst or typical cases and using the same form as in Figures 1 through 7.

6. Change the standard so the double DAV transition will not cause errors. Of the solutions listed in this report, the linear Topology/Schmit Trigger Receivers might be the least painful.

7. Change the standard to require at least four-fifths the devices be powered on.
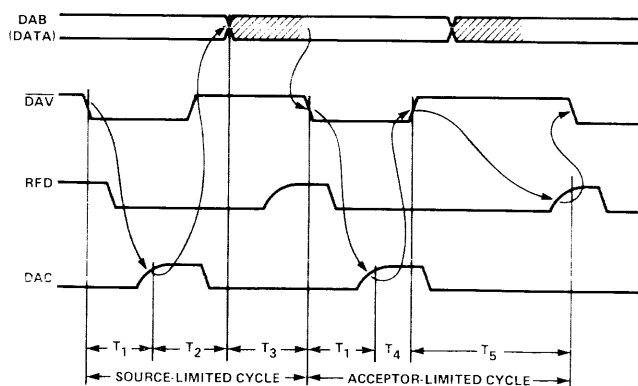
HP-IB Data Handshakes
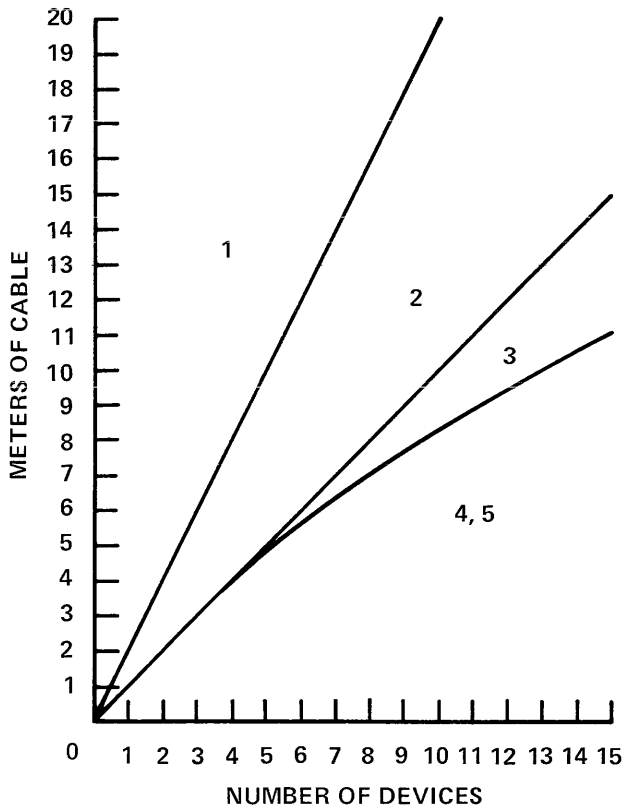
Figure 1. Zero nsec Total Device Delays
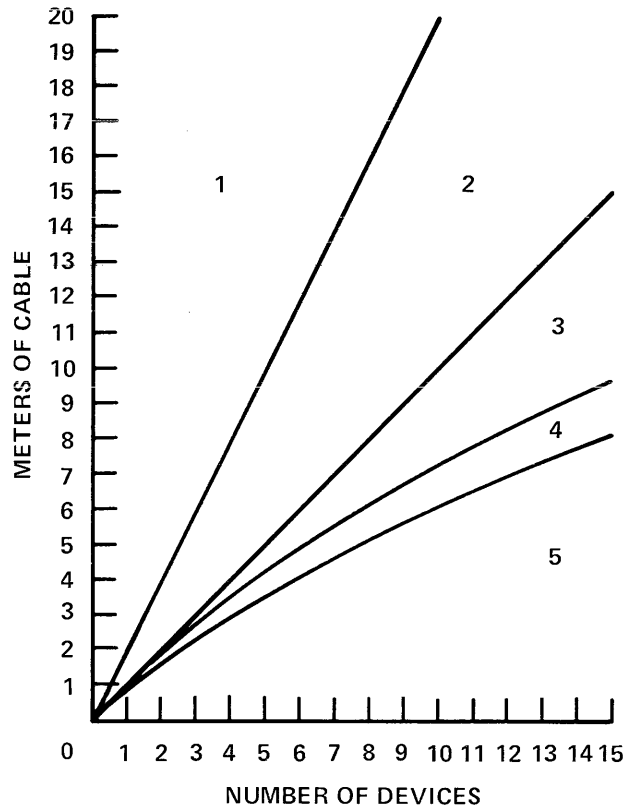
Figure 3. 200 nsec Total Device Delays
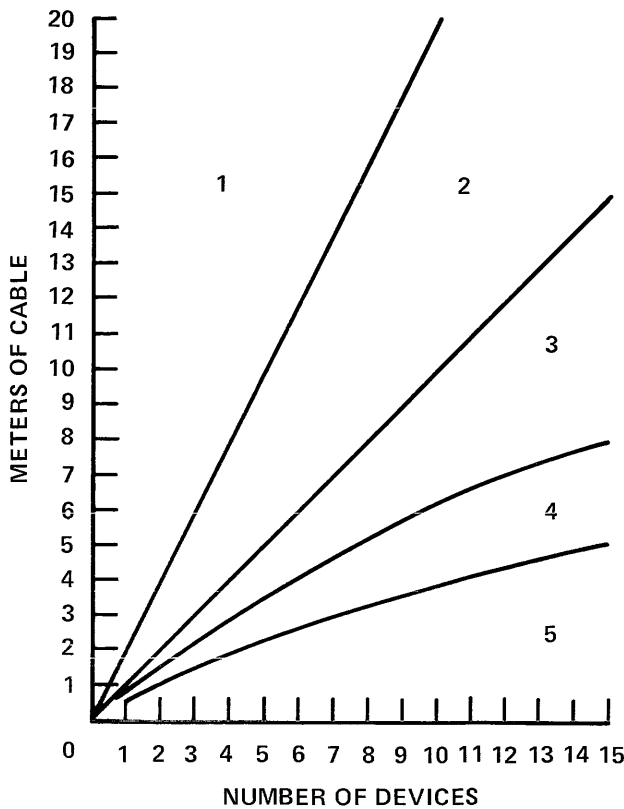
Figure 2. 100 nsec Total Device Delays
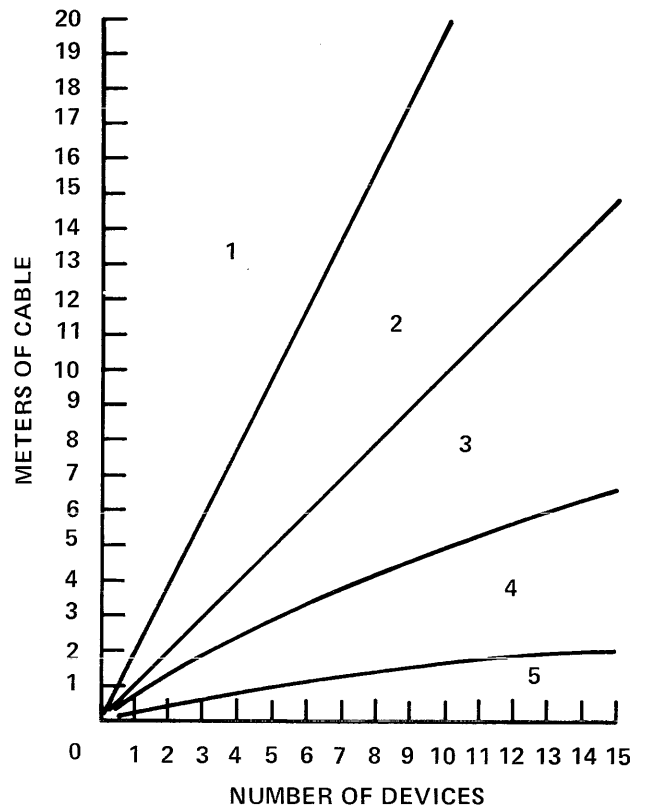
Figure 4. 300 nsec Total Device Delays

about the HP 1000
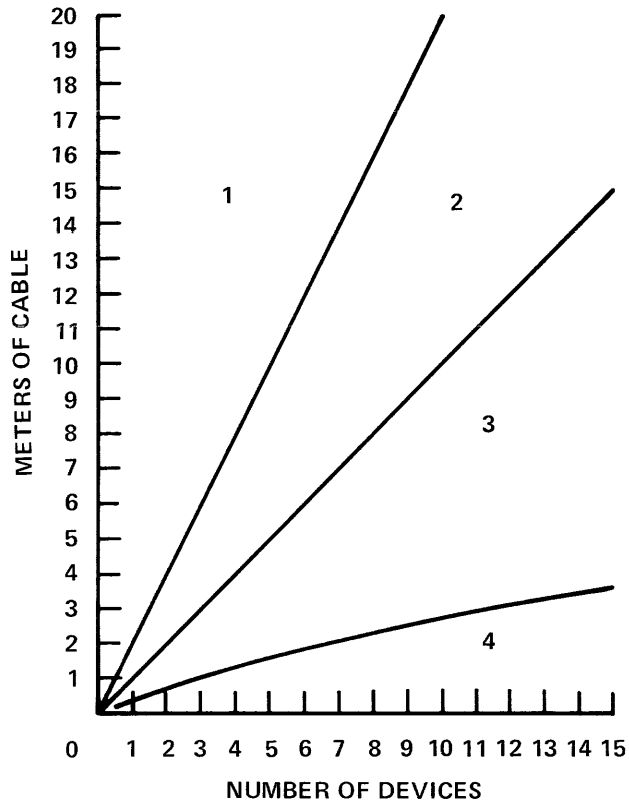
Figure 5. 400 nsec Total Device Delays
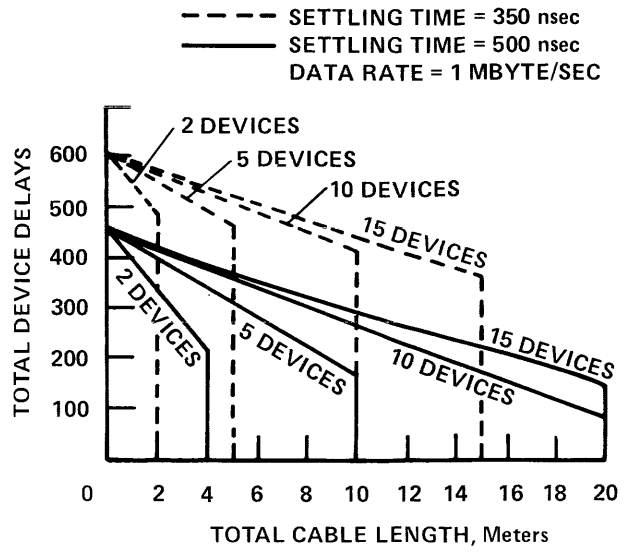
Figure 7. Typical Conditions
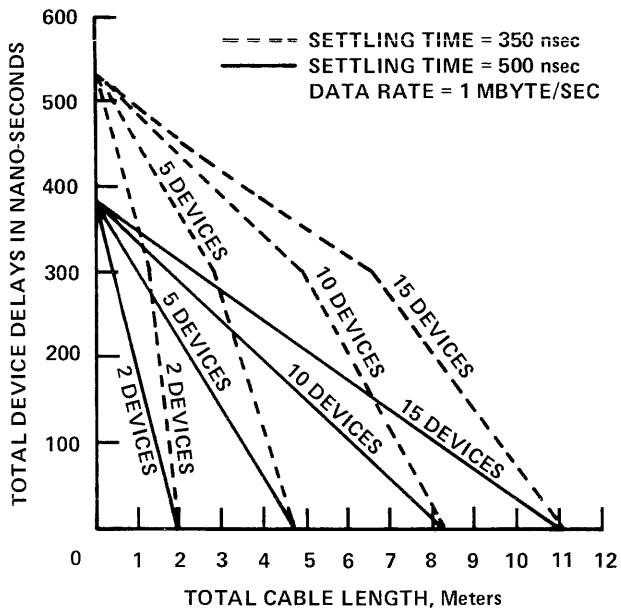
Figure 6. Worst-Case Conditions

about the HP 1000
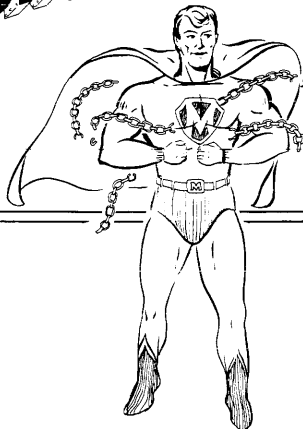
WHAT TO MICROPROGRAM: NEW CONTRIBUTED LIBRARY PROGRAM HELPS DECIDE

Bill Elmore
HP Data Systems

With the availability of HP's new microprogramming package, one is perhaps tempted to microprogram everything. Of course, this would be an enormous and impractical task to tackle. For example, it would be very inefficient to microcode a seldom used library routine; the overall increase in performance would be negligible. An important question then becomes, "What should be microprogrammed?"

It turns out that in many applications where an increase in execution speed is desired, there are particular sections of code, perhaps a subroutine or program loop, in which the program spends a disproportionate amount of time. These sections of code are ideal candidates for microprogramming because, by microcoding a relatively small portion of code, a relatively large increase in performance may be achieved. Therefore, one needs to be able to analyze the program under consideration to determine where these "high activity" regions are located.

A program that does just this, the RTE II/III Activity Profile Generator (APG) for 21MX M/E Computers, has been placed in the contributed library (#22682-18942). APG can be used in an RTE II or RTE III environment on the HP 21MX E- and M-Series Computers. The user inputs program parameters (console and list devices, programs to be analyzed, areas within each program to be analyzed, length of time over which to generate a profile) in a conversational initialization routine, APGI, as shown in the example listing. Specific instructions for use and examples of the available program options are given in the program documentation. APGI then schedules APG.

APG works by sampling the P-register every time the Time Base Generator interrupts, then reducing this data to create an accurate profile of program activity. To reduce the overhead of the APG program, the data gathering portion of this program has been microcoded. To run APG, the user must have a WCS card installed and DVR36 configured into his system. The data reduction portion of APG is scheduled once a second to reduce data until the profile generation has ended. Then control is passed to another program, APGL, which outputs the calculated data.

Following is an example of APG operation and output. Note that two programs are being analyzed, and that each has an obvious area of high activity. Since the data gathering portion of APG is microcoded, the overhead of APG is relatively small, on the order of 2% when it is locked into memory. This results in a very accurate profile.

Note the following considerations when using APG:

1. APG modifies RTE, so any OF,APG command will crash the system. The RTE operator BR command, BR,APG, will allow the user to gracefully exit the program.

2. APG needs two unused base page memory locations. This version of APG uses locations 2 and 3, so if these are not available, two suitable locations must be provided, and appropriate changes made in the program.

3. APG can analyze any number of programs, limited only by a table in which it stores reduced data. To increase the number of programs that can be analyzed, one need only increase the table size.

about the HP 1000

CONSOLE RUN SHEET FOR APG

(operator responses are underlined)

```
          ┌─ CONSOLE LU
          │┌─ APG RUN TIME IN MINUTES
          ││┌─ LIST LU
          │││┌─ WCS LU
          ││││┌─ LOCK APG IN MEMORY
*RU,APGI,1,2,6,13,YE
```

ENTER PROGRAM NAME, FIRST, LAST, DELTA. (0=END)
SINES

ENTER PROGRAM NAME, FIRST, LAST, DELTA. (0=END)
TRIG

ENTER PROGRAM NAME, FIRST, LAST, DELTA. (0=END)
0

START OF ACTIVITY PROFILE GENERATION

*RU,TRIG ◄─────────────── EXECUTE PROGRAMS
*RU,SINES,1 ◄──────────── TO BE ANALYZED
```
TRIG  : STOP        0000
   8    7    41    93
   8    7    29    74
SINES : STOP        0000
```

*BR,APG ◄─────────────── USED TO GRACEFULLY
                          EXIT APG

END OF ACTIVITY PROFILE GENERATION
   APGL : STOP   0000

APG OUTPUT

ACTIVITY PROFILE GENERATION FOR SINES

MEMORY ANALYZED FROM 50000B TO 55643B.
   51 MEMORY CELLS, EACH 00073B LONG;
0 ACTIVITY CELLS NOT LISTED.

LOCATIONS	INTERRUPTS	% USAGE	CUMULATIVE % USAGE
UNDER	.	.00	.00
50000-50072	52.	4.29	4.29
50447-50541	1.	.08	4.37
52046-52140	1.	.08	4.46
52422-52514	1.	.08	4.54
52515-52607	1.	.08	4.62
53540-53632	1.	.08	4.70
55420-55512	190.	15.68	20.38
55513-55605	965.	79.62	100.00
OVER	.	.00	100.00

ACTIVITY PROFILE GENERATION FOR TRIG

MEMORY ANALYZED FROM 50000B TO 51374B.
   51 MEMORY CELLS. EACH 00017B LONG;
0 ACTIVITY CELLS NOT LISTED.

LOCATIONS	INTERRUPTS	% USAGE	CUMULATIVE % USAGE
UNDER	.	.00	.00
50000-50016	80.	2.28	2.28
50036-50054	107.	3.06	5.34
50341-50357	1.	.03	5.37
50606-50624	141.	4.03	9.39
50625-50643	112.	3.20	12.59
50644-50662	207.	5.91	18.50
50663-50701	56.	1.60	20.10
50702-50720	4.	.11	20.22
50757-50775	172.	4.91	25.13
50776-51014	148.	4.23	29.35
51015-51033	316.	9.02	38.38
51034-51052	335.	9.57	47.94
51053-51071	146.	4.17	52.11
51072-51110	48.	1.37	53.48
51130-51146	72.	2.06	55.54
51147-51165	56.	1.60	57.14
51166-51204	71.	2.03	59.17
51205-51223	609.	17.39	76.56
51224-51242	622.	17.76	94.32
51262-51300	54.	1.54	95.86
51301-51317	30.	.86	96.72
51320-51336	89.	2.54	99.26
51337-51355	7.	.20	99.46
51356-51374	19.	.54	100.00
OVER	.	.00	100.00

TOTAL SAMPLES = 4714.
EXECUTION TIME = 47.14 SECONDS

PROGRAM	RELATIVE CPU % UTILIZATION	SAMPLES
SINES	25.71	1212.
TRIG	74.29	3502.

START PRESSING THOSE SOFT KEYS!!!

Gary Gubitz
HP Data Systems

The HP 1000 Computer System has many new features which were described in the feature article of the November-December issue of the **Communicator.** In this issue, let's take a closer look at the "user-programmable soft keys" to see how they can make routine or repetitive keyboarding operations easier, faster, and more accurate.

The "soft" keys are the programmable SPECIAL FUNC-TION keys, f1 through f8, on the HP 2645 Display Station keyboard. A FORTRAN program called "JKEYS" and an associated subroutine called "COLON" simplify the task of programming these keys.

Sample JKEYS prompts and responses are illustrated in the accompanying flowchart and examples. As you enter your response to each prompt, JKEYS builds a file containing the values and labels you wish assigned to the eight SPECIAL FUNCTION keys. After the file has been created, simply dump it to any HP 2645 Display Station with the following command:

:DU, FILENAME, LU

The result? The SPECIAL FUNCTION keys at the Display Station are now programmed for the values that you assigned, and displayed across the top of the screen for the labels for each key. Easy? You bet!!

Example 1 shows the dialog involved in creating a new file containing the following labels and values:

Key	Label	Value
f1	FORMS PROG	RUN FORMS
f2	PART #	12966A
f3	PART #	13197A
f4	PART #	2026A
f5	PART #	2645
f6	UPDATE	YES
f7	STOP FORMS	EXIT
f8	HOME	TR, MENU

Example 2 shows how the file was updated to correct the label and value for key f6 which was erroneously entered as "UPDATE" and "YES" when the file was created.

When the example file is dumped to the Display Station, keys f1 and f7 will be programmed to run and exit from the "FORMS" program. Keys f2 through f6 will store various input values that an operator must respond with as the FORMS program executes.

The real power and potential of the soft key feature would be utilized if each key was assigned a FMGR command that would execute a transfer file. This transfer file could contain a command to dump a different "JKEYS" created file to the terminal. The keys would then be dynamically reprogrammed!! If one (or more) of these new values was a command to also execute a transfer file that contained an instruction to dump even another "JKEYS" created file, the keys could be dynamically reprogrammed again!!!

One might even set up soft key conventions such as reserving key f8 for returning to some predetermined original values.

Actually, this was the exact technique utilized for simplifying demonstrations when the HP 1000 Computer System was introduced. This helped our salesmen a great deal to successfully run the demos!!

Each key was initially programmed such that it represented one particular demo. When any one of the keys was pressed, a transfer file was executed which reprogrammed the keys for that particular demo, and dumped an information file on the screen describing the objectives and operation of the demo. Once the demo was completed, a simple press of key f8 caused a file to be executed such that it returned the original values of the keys (one for each demo available), and displayed a list of the demos available for viewing.

All this power is available to your fingertips. And to help you put it there, the JKEYS program is available from the Contributed Library for your convenience. (22682-18943 see Bulletins). Think of the keystrokes you'll save. And inexperienced operators no longer have to get tangled up in command syntax in order to execute ordinary application programs.

I'm sure you'll agree, that with a little imagination, the possible applications and benefits of the "soft key" feature are endless. So, if you have ideas or suggestions, please let me know so that I can pass them on to all of our readers. Any good porgramming ideas you may want to share can be submitted to the Contributed Library.

So good luck, and start pressing those soft keys!!!

EXAMPLE 1   (User responses are underlined)

RU,JKEYS
JKEYS: VERSION 10-4-76 JRT
JKEYS: CREATE NEW KEYS FILE OR UPDATE OLD?
(CR/UP); CR
JKEYS: ENTER KEY NUMBER (0, WR, EN = WRITE
TO FILE): 1
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
FORMS PROG
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

RUN,FORMS
JKEYS: ENTER KEY NUMBER (0, WR,EN = WRITE
TO FILE): 2
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS);
PART #
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

12966A
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
FILE): 3
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS);
PART #

JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

13197H
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 4
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         PART #
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

2026A
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 5
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         PART #
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

2645
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 6
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         UPDATE
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

YES
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 7
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         STOP FORMS
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

EXIT
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 8
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         HOME
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

TR,MENU
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 0
JKEYS: ENTER FILE 'NAMR' (NAME:SC:CR) FOR
         KEYS FILE/A = ABORT, NULL = REPLACE
         OLD VERSION
         ENTER 'NAMR': KEYSGG:RT;23
JKEYS: END!

EXAMPLE 2

RU,JKEYS
JKEYS: VERSION 10-4-76 JRT
JKEYS: CREATE NEW KEYS FILE OR UPDATE OLD?
         (CR/UP): UP
JKEYS: ENTER 'NAMR' (NAME:SC:CR) FOR OLD
         FILE: KEYSGG:RT:23
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): 6

PART #   UPDATE   STOP FORMS   HOME

4L YES
JKEYS: ENTER KEY LABEL (TO 10 CHARACTERS):
         PART #
JKEYS: ENTER ASCII STRING FOR KEY FUNCTION:

7970B
JKEYS: ENTER KEY NUMBER (0,WR,EN = WRITE TO
         FILE): WR
JKEYS: ENTER FILE 'NAMR' (NAME:SC:CR) FOR
         KEYS FILE
         /A = ABORT, NULL = REPLACE OLD
         VERSION
JKEYS: ENTER 'NAMR':
         END!
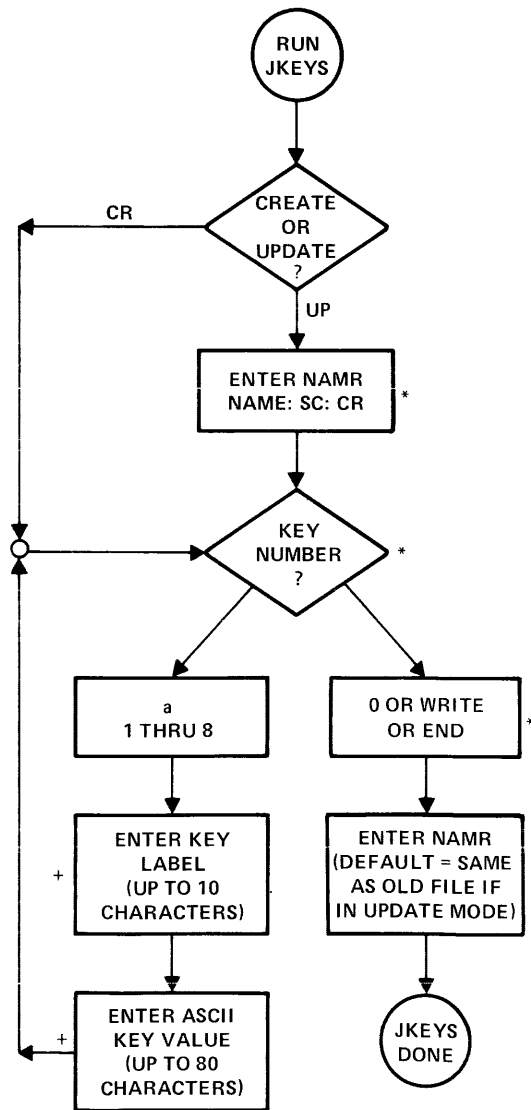
NOTES:

+   Space (considered the default) = no charge if in
      update mode

*   /A will cause termination and clean up.

HINTS ON EFFECTIVE USE OF RTE SOFTWARE

Jim Bridges
HP Data Systems

In the last issue of the **Communicator,** several techniques were presented which help to make more effective use of the system software. This issue will present more ideas on this subject.

Since the last issue, there have been changes to the system itself which eliminate the need for the SYSON program discussed in the previous article. First, FMGR is automatically scheduled at boot-up and control is given to a file named WELCOM. Second, FMGR has been changed to allow it to accept and pass system commands. This is done by preceding commands with the letters SY. For example:

```
:DP, THIS IS A SAMPLE WELCOM FILE
:SYOF,SYSON
:SYON,START
:EX
```

This example suggests that the operator who generated the system forgot that SYSON was no longer needed. But he corrected his error by aborting SYSON in his WELCOM file. Then he scheduled the program START, which (perhaps) was previously scheduled by SYSON.

A number of other changes were made to the system and to FMGR. Also, the system software is now distributed on disc cartridges (7900A or 7905A removable platter) rather than on paper tape. The disc cartridge contains a minimal system which can boot up on a configuration selected by entry into the switch register. The system on the disc contains an on-line system generator (which can be used to generate a new system) and a file area which contains the relocatable modules needed for input to the on-line generator. With the on-line generator, an existing system can be used to generate a new system while the existing system is being used. The output of the generator goes to a type 1 file. A separate program (with its own internal disc drivers) is used to install the new system.

Because of the many changes, the product numbers have been changed for RTE II (from 92001A to 92001B) and RTE III (from 92060A to 92060B). The Batch Spool Monitor (92002A) is now standard (not optional) with RTE II. Existing customers who have purchased a Software Subscription Service for 92001A or 92060A will receive the new software automatically. Customers who have purchased 92001A or 92060A but have not purchased a Subscription Service contract are entitled to purchase the new software at a substantial discount. Please consult your local HP office for more information.

Those of you who adopted the suggestions given last issue for establishing programs added on-line as type 6 files are now (perhaps) wondering how to make regeneration of a system easier. If there are many type 6 files in use, it has probably occurred to you that the process of recreating these files in the new system will be tedious: every program must be loaded, saved in a file and the ID segment must be released. This process could take longer than regenerating the system itself. The technique of dealing with this problem is to automate it by means of a file called *BUILD. The *BUILD file, the relocatables needed for a generation and the answer file to control the generator are stored and maintained on a separate disc cartridge (or mag tape back-up of disc cartridge).

Assume that the *BUILD file is on cartridge number 15. The following is a sample *BUILD file to create type 6 files and set-up and clean up files on cartridge number 2 (assumed to be the system disc, logical unit 2):

```
:LG,20
:ST,/FTN4::15,/FTN4:RT:2
:ST,\FTN4::15,\FTN4:PT:2
:MR,%FTN4
:RU,LOADR,99,6,0,,3
:SP,FTN4
:SP,F4.0
:SP,F4.1
:SP,F4.2
:SP,F4.3
:OF,FTN4
:OF,F4.0
:OF,F4.1
:OF,F4.2
:OF,F4.3
::*LOAD,%SDLS4,%GETRC,0,0,0,0,0,0,(
::*LOAD,%JSAVE,0,0,0,0,0,0,0
::*LOAD,%JSAVE,0,0,0,0,0,0,0
:EX
```

The file *LOAD is given below:

```
:LG,5
:DP,FILE=,1G
:IF,1G,EQ,0,17
:MR,1G
:IF,2G,EQ,0,11
:MR,2G
:IF,3G,EQ,0,9
:MR,3G
:IF,4G,EQ,0,7
:MR,4G
:IF,5G,EQ,0,5
:MR,5G
:IF,6G,EQ,0,3
:MR,6G
:IF,7G,EQ,0,1
:MR,7G
:PU,LOADR,99,6,0,,3
:SP,10G:RT:2
:OF,10G
::
```

Note that the file *LOAD assumes the system is RTE II or that the default can be used for partition and partition size and that the program is not segmented. If the system is RTE III, the maximum partition size could be requested from the operator and stored in a global parameter. This parameter could be used for programs like the FORTRAN compiler, which used buffer area beyond memory declared in FTN4 itself. For example:

```
:DP,TYPE COLON FOLLOWED BY COMMA AND
NUMBER (E.G. :,12)
:DP,WHAT IS LARGEST PARTITION DEFINED
FOR BG PARTITION ?
:PA,0G, ANS =
:CA,8,1G*1000
```

In this case the global 8G is to be used when loading programs that require buffer area beyond declared memory. Thus, the command would be:

```
:RU,LOADR,99,6,,8G,3
```

for these programs. If the program were also segmented (as is the case for FTN4) then global 8G would have to be modified as follows before

```
:CA,8,8G,+,1
```

These examples only hint at the complexity of the *BUILD file, which will be very specific to the individual installation. However, once the *BUILD file (and any auxiliary files needed) are created, it is a relatively simple job to maintain it (them). Thus, regeneration of a system is changed from a tedious to a somewhat pleasant operation.

7905 DISC I/O OPTIMIZATION

Mike Manley
HP Data Systems

System performance on RTE with a 7905 disc can be greatly improved by placing the system disc, LU2, and the auxiliary disc, LU3 (if used) physically in the middle of the disc.

The basic advantage is that head movement (especially in a busy system) can be reduced by more than half. Consider the simple case of the FMGR:MR or :MS commands. These commands generally cause data to come from a peripheral cartridge to LU2 or LU3. When one considers that FMGR

moves data 128 words at a time meaning one head move every 128 words, it pays to keep the distance at a minimum. By having LU2 and LU3 close to the peripheral disc head movement is reduced. This distance will always be at a minimum by placing LU 2 and LU3 in the middle of the disc.

The figure below shows part of an actual generation. The 7905 is divided up into 6 logical cartridges of 203 tracks and one cartridge of 1 track covering 3 surfaces. This division was chosen so that the 7905 would logically look like three 7900 discs and thus preserve backward compatibility with the 7900 disc. Subchannel 0 is LU2, and subchannel 4 is LU3. By looking at the 7905 mapping, the user can easily see the reduced head movement and consequent improvement in system performance.

One problem does result with this configuration. That is, the disc loader ROM won't work. However, the simple FORTRAN program below will correct this.

This problem results because the system boot resides on the first 128 words of LU2 but if LU2 isn't at the physical edge of the disc the ROM is non-functional. This program transfers the first 128 words of LU2 to the edge of the disc. The edge has been saved as subchannel 6 and will only be used for the boot.

Temporarily assign an LU to subchannel 6 and then:

```
RU, BOOT, LU, TRK
```

where

```
LU = the temporary LU

TRK = 0
```

Don't give subchannel 6 a permanent LU (why take chances) and oh yes, don't forget to tell the generator to punch a boot. How else would you get the system up the first time.

One last hint, disc head movement can be reduced even further by adding fewer programs in at generation time and more on-line after generation. Programs like FTN4, ASMB, EDITR etc can be added on-line as temporary loads and saved as files. This will tend to make the heads longer in the file area of LU2 and reduce head movement even more.

about the HP 1000

```
*ANS   T=00003 IS ON CR00016 USING 00024 BLKS R=0000

0001   **   LAB EDIT SYSTEM GENERATION   MIKE MANLEY  10 1 76
0002   17
0003   * #TRKS     STRT CYL    STRT HEAD    #SURF      UNIT#     # SPARES
0004   *
0005   203,       103,         0,          2,         0,         3
0006   203,         1,         0,          2,         0,         1
0007   203,       206,         0,          2,         0,         1
0008   203,       308,         0,          2,         0,         3
0009   203,         1,         2,          1,         0,         2
0010   203,       206,         2,          1,         0,         2
0011
0012
0013     3,         0,         0,          3,         0,         0
0014   *


0037   *      7905 SUBCHANNEL MAPING
0038   *         CYLINDER'S:
0039   *    *0 1    103    206    308    411
0040   *HEAD*********************************
0041   * 0 *!*   !   *   !   *   !   *   !   *
0042   *   *6*--1--*--0--*--2--*--3--*
0043   * 1 *!*   !   *   !   *   !   *   !   *
0044   *   *!***************************
0045   *   *!                                *
0046   *   *!***************************
0047   * 3 *!*     4      *     5      *
0048   *   *-*-----------------------*
0049   * 4 *X*            *            *
0050   *   ***************************
0051   *
0052   *
```

Upper platter

Lower platter

```
%BOOT  T=00004 IS ON CR00003 USING 00002 BLKS R=0009

0001   FTN,L
0002        PROGRAM BOOT
0003        DIMENSION IB(128),ILU(5)
0004        CALL RMPAR(ILU)
0005        CALL EXEC(1,2,IB,128,0,0)
0006        CALL EXEC(2,ILU(1),IB,128,ILU(2),0)
0007        STOP
0008        END
0009        END$
```

about the HP 1000

KNOW YOUR RTE - PART 6

by Mr. RTE

Last time we discussed an I/O request. This time we will discuss a particular and special I/O operation, i.e. time keeping. If you will recall in a past issue we discussed start up of the system, that is, what happens when the system is booted. At that time we indicated that the time base generator (TBG) was started. Time base interrupts occur each 10 milliseconds (MS) and, like most standard interrupts are vectored to RTIOC at entry point $CIC. As per standard interrupts the interrupt system is turned off and the state of the system is saved. The interrupt is then identified, in this case by comparing the interrupting select code with the contents of location 16478 in the systems communication area of the base page. RTIOC will then go to the entry point $CLCK in the RTIME module.

RTIME code then steps the system time — which is kept as a double integer in 10's of MS. A double integer has more than enough room for 1 full day of 10 MS ticks. One word is stepped, if it goes to zero the second word is stepped. If the second word goes to zero it is midnight and the double integer is reset for the next day while the day word is stepped. The day and year are kept in one word in days with the base year being 1970. That is take the year, subtract 1970 multiply by 365 (the number of days in a standard year) and add the current day of the year and then subtract one more and you have the contents of the day/year word. The additional decrement represents the fact that there is no day 0. (The absence of a day 0 is a common artifact of our numbering system which also shows up, to our consternation, in a multitude of places in programs. Yes Linda, there is a 0!) Of course that day has to be added back again if we need to obtain the correct day, after we divide by 365 to separate the day and year.

After the time update each program in the time list is checked to see if the time to run is now. This is a two word compare. If the time is now and if the program is dormant it is scheduled.

The program's next run time is computed and set each time it should be run even if it can not be run because it is still active. If, during this update, it is found that the multiple is zero then the resolution is also cleared and the program is taken out of the time list.

The time tick processing continues by checking if D.RTR or SMP is the current executing program. If not then, if the current program is running under BATCH (the BA bit is set in his ID Segment) the two word double integer batch time is stepped. This time is set up by the BATCH monitor as part of job processing. If it goes to zero a batch time out condition exists. In this case the system checks to see if the

current program is a son by checking the FW bit in its ID segment. If the program is a son it is aborted with the TO abort message (call to $ERMS). If it is not a son it must be the batch processor so its break flag is set. This is the bit labeled AT in word 21 of the ID segment. The student will observe that this whole batch time processing trip references only word 21 of the ID segment, thus avoiding a lot of indexing into the ID-Segment.

The next operation performed by the time tick processor is to index into each EQT at word 15. If the word is non zero it is incremented. If it goes to zero the EQT has timed out. If this happens the tick processor goes back to RTIOC at $DEVT to do I/O time out thing. $DEVT checks EQT word 4 to see if the driver wants to handle the time out or not. If the system is to handle time out (i.e. the driver has not set the bit in EQT word 4) it forces an entry into the standard I/O completion code with a completion code of 4. This will cause a time out message and will down the LU. If the driver is going to handle the time out then the system simulates an interrupt from that devices select code and enters the device at C.XX. The driver has to figure out that in fact there was a time out by looking at the TM bit in the EQT. RTIOC does not return to the time base processor after handling a time out, thus, the first EQT to time out gives all the higher numbered EQTS a 10 MS. grace period.

If no devices have timed out the tick processor goes to the dispatcher at $XEQ. This completes the discussion of time tick processing, which is summarized in Figure 1.

We will now go through the EXEC 6 request and its options. Last time (know your RTE part 5) we did an EXEC I/O call where we discussed the front end processing that every EXEC request goes through. So we start the EXEC 6 discussion at the entry point $MPT1 in SCHED. This is the code which processes the EXEC 6 request. The first thing done at this point is to figure out the ID segment address for the program to be acted upon. The EXEC 6 requests' parameter 2 indicates which program this is by naming the program or, by being 0 or absent, indicating that the caller is to be suspended. After the ID segment is identified (in either way) if it is not the current program a parentry test is made.

If the program is not a son of the caller, the caller is aborted with an SC04 via the $ERMS routine. The parenthood test is done with the father field in word 21 of the referenced program. If the test passes, the calling programs point of suspension is advanced and the termination options are checked. The checks are made in numerical order so, first, for −1 (or serially reusable termination): If the terminating program is not the current executing program a standard 0 option termination is done (a program is not assumed to be serially reusable unless it says it is — its fathers word is not enough). Otherwise the routine TERM (which we discussed in connection with the

584

OF request) is called to put the program dormant etc. and then the least bit of word 21 (the father pointer bit 0) is set.

The call to $LIST that TERM made put the program in the list headed at $ZZZZ in the DISPA module. When DISPA processes this list it will see the bit set in word 21 and will not clear the programs ownership information on the partition it currently resides in. If the partition is needed for some other program, however, this programs point of suspension being zero will cause DISPA to overlay it, rather than swap it. On the other hand, if the partition is not needed when this program again becomes scheduled it will be run from its primary entry point without reloading from the disc.

After the bit is set in word 21, SCHED finishes the EXEC 6 processing by passing any optional parameters to the program and then jumping to $XEQ.

For the 0 (standard termination) option the system calls TERM and then passes any parameters and goes to $XEQ. For the 1 (save resources termination) option the system sets the R bit (save resources bit) in the programs ID Segment and if a son is being suspended calls $LIST to put the program in the dormant list and exits to $XEQ. If the current executing program made the request then any programs waiting for this one to complete are scheduled by calling the $WATR subroutine and then TERM is called, parameters passed and control transferred to $XEQ. $WATR must be called here in this case as DISPA usually calls it while processing the $ZZZZ list but the program is not put in this list because the R bit is set. On the other hand $WATR is not always called at this point because if the father is suspending his son, the son might be in I/O suspend status and the call to $LIST will not make him

dormant but will set the D bit instead. The program will then go dormant when the I/O completes. At this time $LIST puts him in the $ZZZZ list and subsequently DISPA will call $WATR to pick up any programs waiting for him. If $WATR has been called too early the programs might have been put right back in the wait state and we would still need the call in DISPA.

For the 2 (soft abort termination) option $MPT1 transfers control to the OF routine as if an OF, *name* had been entered. For the 3-or hard abort termination $MPT1 transfers control to the OF routine as if a OF, *name,* 1 had been entered. This will cause a *name* ABORTED message to appear on the system console.

With this issue we conclude the discussion of the system code. Next issue we will discuss the power fail auto restart option. If there are any questions or items you would like to know more about please write. Current plans are to discuss FMGR, batch, spool and the generation procedure (why should I have answered that generation question differently?) Beyond that we have no plans so let us know what you want to hear.

The figure below shows all major entry-points of the real-time clock-processor $CLCK, which handles all time-dependent functions in the RTE.

The processing of system-interrupts is controlled by directing all sources to the entry-point $CIC.

In the case of a TRG-interrupt, the A-register gets the select-code of the TBG-card from the central-interrupt-register and the comparison with the contents of address TBG (=1674) holds true. This causes a jump to $CLCK.

ENTRY POINTS				REMARKS
$CLCK				STEP SYSTEM TIME
CLØ1Ø				GO TO PROCESS LISTS
CLØ11				CHECK TIME LIST FOR SCHEDULING PROGRAMS THRU PROCESSING TIME LIST < ID(17) >
	TMSCH			SCHEDULE THE PROGRAM
	$LIST			IF DORMANT
	TUDAT			UPDATE THE SCHEDULE-TIME
	TOBAT			PROCESS BATCH TIME-OUT
		ABOR		
			$ERMG	ERROR MESSAGE
			$SYMG	SYSTEM MESSAGE: TI
			$ABRT	ABORT
		IDTOP		ID TIME OUT PROCESSOR
		$DEVT		GO TO TIME-OUT PROCESSOR
		$XEQ		

Figure 1.

Software Samantha

about the HP 1000

```
        bit
word     15                                    0
        ------------------------------------------
     -                                           -
  1  -  F                        P P P P C C C C  -
     -                                           -
  2  -  1st character    2nd character            -
     -                                           -
  3  -  3rd character    etc.                     -
     -                                           -
        ------------------------------------------

     F    =  1 if function, 0 if subroutine
     PPPP =  number of parameters
     CCCC =  number of characters in name
```

Table I. Format of Mnemonic Table Entry

Dear Samantha,

How does BASIC convert and pass variables and arrays to FTN or ASSY subroutines? What limits the size of the passed array?

Bill Otto
DFSMI-RRS
Army Missile Command
Huntsville, Alabama

BASIC makes extensive use of the branch and mnemonic tables in the execution of the call statement. The table generator, RTETG, creates the branch and mnemonic tables, as type 7 files and the transfer file as type 3 from the information the user has supplied in the command input file. The transfer file is used to load the overlays as permanent programs under the name %BANN, when NN is the overlay number. When BASIC encounters the TABLES command the two type 7 files are opened and read into memory.

The mnemonic table contains the name of the subroutine, the number of characters in the name, a flag bit indicating function or subroutine, and the number of parameters in the call. Table I illustrates the format for the mnemonic table entry.

From the branch table BASIC receives the location of the routine, and three pieces of information about each parameter in bits 0 through 14 of the next three words, the type of the parameter, whether it is passed to the subroutine or returned from the routine, and if conversion to integer is required. Table II illustrates the branch table format.

```
        bit
word     15                                    0
        ------------------------------------------
     -                                           -
  1  -  D D D D D P P P P P S S S S S S           -
     -                                           -
  2  -  X A A A A A A A A A A A A A A A           -
     -                                           -
  3  -  X T T T T T T T T T T T T T T T           -
     -                                           -
  4  -  F I I I I I I I I I I I I I I I           -
     -                                           -
        ------------------------------------------

     DDDD  =  identification number
     PPPP  =  overlay number
     SSSSS =  subroutine number within overlay
     A     =  1 if array, 0 if non-array
     T     =  1 if from subroutine, 0 if to subroutine
     F     =  1 if integer function, 0 if real function
     I     =  1 if conversion to integer required,
              0 if no  conversion required
```

Table II. Format of Branch Table Entry

BASIC first checks to see if executing in simulate mode. If so, control is transferred to segment seven of BASIC to simulate the call. Else, a three word descriptor is built for each parameter and the descriptor triplets are stacked on a run time stack in memory in reverse order followed by the parameter count on the top of the stack.

```
        parameter
word    simple variables,    arrays           strings
        ---------------------------------------------------
     -                      -             -                -
  1  -  argument pointer    -  element pointer  -  -base address - 1 -
     -                      -             -                -
  2  -  argument pointer    -  array base ptr   -  character address -
     -                      -             -                -
  3  -  2:real / 1:into     -  array size(wds)  -  -strg lngth(char) -
        ---------------------------------------------------
```

Table III. Format of Descriptor Triplet

For FORTRAN functions the descriptor block is built and stacked, but the call number and parameter count are placed on a second stack. A temporary stack is used to hold intermediate results from execution of the function.

Next the subroutine number is used to find the branch table entry from which the control word and parameter conversion words are retrieved. The control word is used to build the name of the overlay. BASIC then writes out the descriptor block and the parameter count to system available memory (SAM) using class I/O, followed by a class write for each parameter. At this point the parameter values are converted according to the contents of the parameter conversion words. These three words define the type of variable the subroutine is expecting to receive and the direction the variable is passed.

Upon return from the subroutine the parameters are reconverted as defined by Table IV above and written to SAM through class I/O. Control is transferred back to BASIC and the values are picked up from SAM if the return flag is set for that parameter and if no error was generated in the process.

What limits the size of a passed array? There are no restrictions in BASIC which limit the array size except it is limited to the largest integer value which can be stored in one word. The restrictions come in the availability of SAM. The entire array is passed as one record through class I/O and the descriptor block and entire set of parameters are stored in free core in the partition in which the overlay is located. Thus one must be careful in assuring the availability of ample free core.

formal argument: actual argument:	array/simple	return/no	integer/real
simple variabe	pass variable	save return if bit = 1	fix on call and float on return
array variable	pass array if 1 with pointer to given element pass element if 0	pass value(s) save return(s) if bit = 1	fix all values and float on return
string	pass string or pass 2 chars if 0	save string or	
string constant	pass string or substring if 1 pass 2 chars	syntax error if bit = 1 pass only if 0	
simple constant	pass constant	syntax error if bit = 1 pass only if bit = 0	fix on call and float on return if bit = 1
real expression	syntax error if bit = 1	syntax error if bit = 1	fix value on call

Table IV. Parameter Conversion

BASIC now schedules the program %BANN, when NN is the overlay number. A class get is used to pick up the descriptor block and parameter count followed by a class get for each of the parameters. CALSB instructs the system to swap the entire disc resident area if swapping is necessary. This allows the routine to use undeclared core to store the descriptor block and parameter values. A subroutine call is built with the addressess of the parameters in the overlay, and the call is executed.

about the HP 1000

bulletins

EVERYTHING YOU WANT TO KNOW ABOUT MICRO-PROGRAMMING YOUR 21MX M-SERIES COMPUTER (AND YOU WON'T HAVE TO BE AFRAID TO ASK!!)

Paul McGillicuddy
Mark Beswetherick
HP Data Systems

Are you one of our many customers who has heard about the powerful microprogramming capabilities available to users of 21MX M-Series Computers operating under the control of RTE-II/RTE-III, version E or later? If so, have you found yourself wondering: "What are these capabilities and how can I use them? Will they provide me with the tools I need to improve system performance? Does micro-code provide the best solution for every programming problem? If not, how do I decide what should or shouldn't be microprogrammed?"

You will get answers to these and other questions you may want to ask by attending our microprogramming class (22960A). This compact, five-day course is designed to bring you up-to-speed in a hurry. It fully explains the tools associated with microprogramming and shows students how to use them to write their own microprograms.

Lab sessions, which comprise almost half of the course, are carefully structured to closely simulate real-life applications. For example, each student microcodes a section of a privileged driver. There is also a problem that allows students to analyze a typical application to determine what sections of a the program are CPU-BOUND. This is accomplished by using a utility called the Activity Profile Generator (APG). Students then microcode the problem areas of the program and analyze the improvement in execution speed. Typical improvements achieved by STUDENT PROGRAMS average five to one!

In addition the Course also covers:

● Microprogramming a high speed sort routine (Shell)

● Dynamic Mapping System

● RTE Microprogramming software

● How to install Microprograms temporarily and/or permanently

● Calling microcoded subroutines from Fortran or Assembler programs.

So if you want to learn more about microprogramming power and its potential to turn your RTE system into a real tiger, why not enroll today?? Microprogramming is an excellent way for the "SOFTWARE" oriented student to get closer to the 21MX architecture, and for the "HARD-WARE" oriented student with machine language experience to gain experience in programming techniques. Potential students should be familiar with RTE and understand assembly language concepts. There are still some openings in classes beginning on the following date:

March 28, 1977

The fastest way to enroll is to contact your local HP Representative. For further information, call the training registrar at (408) 257-7000 EXT. 2952.

NEW RTE MICROPROGRAMMING SOFTWARE

Don Ried
HP Data Systems

RTE, microprograms, and writable control store (WCS) have teamed up for super fast programs! The new HP 92061A RTE Microprogramming Software Package allows on-line development of microprograms in an enhanced RTE-II/-III system (date code 1631). This is a powerful new tool for users who want to reduce execution time of their programs in a 21MX E-/M-Series Computer. The microdebug editor provides complete interactive editing capability using *symbolic* — *not* octal — codes! Editing capability remains "alive" during microprogram breakpoint halts. The driver loads microprograms into WCS for high speed execution. The software also includes a cross-reference generator and a PROM mask tape generator. Two new manuals, *HP 21MX E-Series Computer RTE Micropro-gramming* (part no. 02109-90004) and *HP 21MX M-Series Computer RTE Microprogramming* (part no. 02108-90032), provide complete information on preparing, microassembling, and editing microprograms. (By the way, the new 1K-capacity 13197A WCS Board can hold four times as many microinstructions as previous HP WCS boards and requires less than half the power.)

HP ALGOL REFERENCE MANUAL

Dave Tribby
HP Data Systems

A new edition of *HP ALGOL Reference Manual* (part number 02116-9072) is now available. This edition, dated November, 1976, is not just a revision of the previous edition — it has been completely reorganized and rewritten. Of particular interest are an expanded section on FORMAT

specifications, more information on Assembly Language routines, a new section comparing HP ALGOL and HP FORTRAN IV, and an appendix describing the language in Backus-Naur Form (BNF) productions.

The manual also reflects a recent change in the ALGOL compiler: the object code listing option produces Assembly language mnemonics in addition to octal numbers for each instruction generated by the compiler.

Copies of the manual can be ordered through your local Sales and Service office. The address and telephone number of the office nearest you are listed in the back of all customer manuals. Customers in the U.S. may also order directly by mail. Use the form supplied at the back of the **Communicator.**

HP'S MICROPROGRAMMING BEST-SELLERS

Mark Beswetherick
HP Data Systems

Looking for an opportunity to do some interesting reading as well as learn about microprogramming? Then you should turn to our 21MX M-Series or E-Series Microprogramming Reference Manuals.

These well-written manuals serve as an introduction to Microprogramming concepts as well as a complete reference source for Microprogramming. Topics covered in the manuals include System Architecture, Microassembly Language, and RTE Microprogramming Software. Both manuals are liberally sprinkled with examples, including flow charts, comments, and console run sheets.

Whether you are new to microprogramming or a microprogramming expert, you will find these references invaluable.

To order, contact your local HP representative.

Part Numbers:

HP 21MX M-Series Computer RTE Microprogramming Reference Manual — 2108-90032
HP 21 MX E-Series Computer Microprogramming Reference Manual: 2109-90004

NEW RELEASES FROM THE 2100/21MX CONTRIBUTED LIBRARY

Melanie Van Vliet
HP Data Systems

This article serves as an update for the 2100/21MX Contributed Library Program Catalog (22999-90040).

The new contributed programs listed below are now available. Contact your local HP sales office to order Contributed Library material, or (if you are in the U.S.), you can use the Direct Mail Order form at the back of the **Communicator.**

22682-18931 **USER SPOOL POOL FILE ACCESS OR SPOOLING MADE EASIER**

SPLUM enables a user program to spool a logical unit via a spool pool file, i.e. SPLUM overcomes not being able to utilize the spool pool files outside of batch.

SPOLU is a Fortran callable interface subroutine to SPLUM.

SPLOT is a Fortran callable subroutine to close and pass the spool pool file set up by SPOLU and SPLUM. The use of SPLUM, SPOLU and SPLOT permits easy access to the spool features of RTE by the non-experienced system's programmer. At last all programmers may spool to such things as the line printer to avoid interweaving without having to play games with SMP and creating their own spool files.

22682-18931	PT	$10.00
22682-13331	Cass	$35.00

22682-18932 **"LOAD ON CALL" (LOCAL) PACKAGE**

This package allows each subroutine called by a Fortran main program, to be placed in a segment. Segments are loaded at run time. The first program runs after compilations and before the relocating loader. It asks for the main and subroutine names and after several verifications creates segments in job binary area.

The second program is a subroutine called by the main program. It verifies and assumes argument transfer to the proper subroutine by loading the appropriate segment if non-preceding in core. This package allows appreciable core space and the storage total required is equivalent to the main, the largest subroutine, plus 150/10 words for the "LOAD CALL" routine.

22682-18932	PT	$40.00

22682-18933 **12 X 12 CHARACTER GENERATOR SUBROUTINE**

The 12 X 12 Character Generator allows the user to create block letters from the following set;

ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789$#:"/+!'( )*?.-%] [

The subroutine is called with the desired character, a 12 X 12 matrix, and the desired pattern character. The matrix is then filled in AI format, with the desired character using the pattern character.

22682-18933	PT	$30.00

22682-10934 **RTE MACRO PROCESSOR**

This program is a generalized Macro Processor which provides facilities for defining, invoking, and purging macro-definitions. It may be used to extend existing programming languages by adding macro statements, or to translate similar languages.

22682-10934	800BPI	$40.00
22682-11934	1600BPI	$40.00

22682-18935 **DATEL SYSTEM 256 ANALOG-TO-DIGITAL CONVERTER ROUTINE**

This program is a driver for a multi-channel analog-to-digital converter system; DATEL System 256. It is designed for the case where a set of analog signals have to be sampled over a long period. The amount of data acquired this way is too large to be held in memory, allowing the user to store one buffer on a mass-storage device while the other buffer is being filled. The sampling rate is set by means of a time base generator.

The controller routine consists of an initiation section which initializes the time base generator and a continuation section where the analog signals are sampled and stored in the data buffer.

22682-18935	PT	$10.00

22682-18936 **FAST FOURIER TRANSFORMER WITH DATA-STORAGE ON DISC**

This program finds the forward Fourier transform of a large complex data array stored on disc, using the Cooley-Tukey algorithm.

The program is written for a block size of 4096 but this length (an integer power of two) can easily be changed. The form is executed in place, so the original data are lost. The result is left in bit reverse order, but a separate program for printout in the right order is provided. The transform program uses a second array on disc for storage of sine — and cosine values needed.

22682-18936	PT	$10.00

22682-18937 **DLU FOR RTE**

DLU is a program to dump, restore and verify whole 7900 cartridges to and from magnetic tape. It is compatible with the stand-alone DLU (25123-60030) but it runs under RTE.

22682-18937	PT	$10.00
22682-13337	Cass	$35.00

22682-18938 **DOS-IIIB INTERACTIVE EDITOR**

EDITR is a DOS version of the RTE interactive editor. It allows a user to edit type-S (source) files conversationally through a teleprinter console. Both line and character edits are supported. The user enters a series of control commands to edit the original file. Lines can be inserted or deleted, and text exchanges can be made. Unlike the standard DOS editor (:ED), EDITR will allow the file to be rescanned for further editing. A DOS operating system must be used, because EDITR uses the FILE CREATE, FILE PURGE, and FILE RENAME EXEC calls (EXEC 32, 33, 34). DOS-IIIA and DOS-M will not work unless this program is modified to handle its own directory changes. EDITR uses all available core (exclusive of itself) for buffering the disc, therefore, program efficiency increases if more than the minimum memory requirement is available.

22682-18938	PT	$30.00

about the HP 1000

22682-18939   **BREF — CROSS REFERENCE TABLE GENERATOR FOR HP 92101A**

BREF is a cross-reference table generator for HP 92101A Multi-User Real Time Basic. It will accept a File Manager source file for input and produce a formatted listing of all variables used in the program. The listing is divided into four parts:

1. 1 char simple variables
2. string variables
3. array variables
4. 2 char simple variables

Within each part, the list of the variables is alphabetical, and the list of line numbers which follows each variable is arranged numerically. The algorithm used by BREF is syntax-independent, therefore, BREF should remain compatible with future revisions of RTE-II Basic.

22682-18939	PT	$10.00
22682-13339	Cass	$35.00

22682-18940   **ASCII SOURCE FILE WRITE SUBROUTINE**

The ASCII Source File Write Subroutine enables the user to have the following capabilities:

1. Source File Write Capability
   The user may concurrently write to a maximum of ten disc files. These disc files need not be an ASCII disc file, but if it is, the file write format is compatible with DOS source file structure. Each opened file is assigned a buffer (128 Word minimum). Depending upon the size of the buffer, I/O efficiency may be greatly increased since the routine performs both logical and physical writes.

2. Sequential Write Capability
   As mentioned above, the destination file does not have to be a source file, thus the user may use this subroutine simply as a powerful high speed sequential file write subroutine.

Since the HIGH SPEED DISC INPUT/OUTPUT SUBROUTINE is required (HP# 22681-18981), the user may write on the scratch areas of both the current user disc and the system disc by using the file names $USER and $SYSTM.

22682-18940	PT	$20.00

22682-18941   **DOS-IIIB CHANGE SYSTEM CONSOLE PROGRAM**

This program allows the user to reassign the system console (LU# 1) to any remote terminal.

There are only two restrictions to this program:

1. The remote terminal driver must be core resident.

2. The remote terminal driver must not contain DAM.

Once the remote terminal has been assigned as the system console, the user may use this program to change back to the original terminal.

22682-18941	PT	$10.00

22682-18942   **RTE 2/3 ACTIVITY PROFILE GENERATOR FOR 21MX M/E COMPUTERS**

This contributed program package analyzes activity in 1 or more application programs. On a basis of this activity, the applications may be optimized by recording or microcoding.

The programs being analyzed need not be modified for purposes of the analysis. The analysis program runs in RTE-II or RTE III on a 21MX — M or 21MX — E series.

22682-18942	PT	$80.00
22682-13342	Cass	$70.00

22682-18943   **PROGRAMMING FOR HP 2645 TERMINAL "SOFTKEYS"**

JKEYS operates interactively, prompting and receiving from the operator the data required to create or edit a disc file

about the HP 1000

containing programming information for HP 2645 terminal "Softkeys". "Dumping" the file (":DU," FMGR command) to a 2645, will give these results:

a.   Programs the 8 softkeys as specified.

b.   Writes a "menu" across the top of the CRT screen, in a protected field, using inverse video capabilities of the terminal.

22682-18943	PT	$10.00
22682-13343	Cass	$35.00

22682-18944   ASCII SOURCE FILE READ SUBROUTINE

The ASCII Source File READ Subroutine enables the user to have the following capabilities:

1.   Source File READ Capability;
   The user may concurrently read to a maximum of ten disc files. These disc files need not be an ASCII disc file, but the file being read must be structured the same as an ASCII disc file. Each opened file is assigned a buffer (128 words minimum). Depending upon the size of the buffer I/O efficiency may be greatly increased since the routine performs both logical and physical reads.

2.   Sequential READ Capability;
   As mentioned above, the input file does not have to be a source file, but the file structure must be identical to a source file. The user may use this subroutine simply as a powerful high speed sequential file read subroutine.

Since the High Speed Disc Input/Output Subroutine is required (HP# 22681-18981), the user may read the scratch areas of both the current user disc and the system disc by using the file names $USER and SSYSM. (These files must comply to the ASCII file structure also).

22681-18944   PT $10.00

EDITOR'S NOTE

Gary Gubitz
HP Data Systems

Correction to "FORTRAN IV I/O" using "ASSIGN" statements.

It has been brought to our attention that a problem existed in the program sequence shown in the **Communicator** (pg. 480, issue #10, date Sept. 76).

Below is a corrected listing:

```
&ASIGN T=00004 IS ON CR00011 USING 00002
   BLKS R=0009

0001   FTN4,M
0002         PROGRAM ASIGN
0003         DIMENSION IX(0), IY(0)
0004   100   FORMAT("THIS IS A HEADING")
0005         ASSIGN 100 TO IB
0006         IY(0)=IB+3
0007         CALL EXEC(2,1,IY(0),9)
0008         END
0009         END$
```

Several problems were noted in using the information as published; an error 22 from FORTRAN IV because the ASSIGN can't reference an array. This was fixed by assigning 100 to IB. Then, to correctly link the addresses, IY(0) is set to IB+3 to get past a JMP & the (". (See mixed listing.)

Hopefully this hasn't caused too many problems. Sorry for the mix-up.

MIXED LISTING

```
PROGRAM ASIGN
     00000     000000     NOP
     00001     016001X    JSB CLRIO
     00002     000003R    DEF *+1
     00003     026022R    JMP 00022
     DIMENSION IX(0), IY(0)
100  FORMAT("THIS IS A HEADING")
     00004     026020R    JMP 00020
     00005     024042     ASC 1,("
     00006     052110     ASC 1,TH
     00007     044523     ASC 1,IS
     00010     020111     ASC 1, I
     00011     051440     ASC 1,S
     00012     040440     ASC 1,A
     00013     044105     ASC 1,HE
     00014     040504     ASC 1,AD
     00015     044516     ASC 1,IN
     00016     043442     ASC 1,G"
     00017     024415     ASC 1,)
```

```
ASSIGN 100 TO IB
    00020      000021R      DEF *+1
    00021      000022R      DEF *+1
    00022      062024R      LDA *+2
    00023      002001       RSS
    00024      000004R      DEF 00004
    00025      072053R      STA IB

IY(0)=IB+3
    00026      062052R      LDA 00052
    00027      042054R      ADA 00054
    00030      042021R      ADA *-7
    00031      072056R      STA A.001
    00032      062055R      LDA 00055
    00033      042053R      ADA IB
    00034      172056R      STA A.001,I
CALL EXEC(2,1,IY(0),9)
    00035      062052R      LDA 00052
    00036      042054R      ADA 00054
    00037      042021R      ADA 00021
    00040      072056R      STA A.001
```

```
CALL EXEC(2,1,IY(0),9)
    00041      016002X      JSB EXEC
    00042      000047R      DEF *+5
    00043      000057R      DEF 00057
    00044      000062R      DEF 00062
    00045      100056R      DEF A.001,I
    00046      000060R      DEF 00060

END
    00047      016002X      JSB EXEC
    00050      000052R      DEF *+2
    00051      000061R      DEF 00061
    00052      000000       OCT 000000
                            BSS 00001
    00054      177777       OCT 177777
    00055      000003       OCT 000003
                            BSS 00001
    00057      000002       OCT 000002
    00060      000011       OCT 000011
    00061      000006       OCT 000006
    00062      000001       OCT 000001
```

software updates

Listed below are the software parts and manuals which are shipped when 92001B (RTE-II) is ordered (or is included in an order).

SOFTWARE MODULE NUMBERS: 92001B

Software Module Number	Module File Name	Module Descriptor	Part Numbers			Date Code or Revision
			Mini-Cartridge	7900 Disc	7905 Disc	
02607-16004	!S4L07	24K SIO LINE PRINTER DRIVER	92001-13305	92001-13001	92001-13101	1538
09601-16021	%DVR15	RTE 7261A DRIVER	92062-13301	92001-13001	92001-13101	A
12970-16004	!S4MT1	24K SIO MAG. TAPE DRIVER	92001-13305	92001-13001	92001-13101	1550
20747-60001	%DVR30	RTE FIXED HEAD DISC DRIVER	92062-13301	92001-13001	92001-13101	C
20808-60001	%CAL10	CAL. PLOTTER DRIVER	92062-13301	92001-13001	92001-13101	B
20810-60001	%CALIB	CAL. PLOTTER LIBRARY	92062-13301	92001-13001	92001-13101	C
20875-60001	%1FTN	FORTRAN MAIN CONTROL	92060-13308	92001-13001	92001-13101	E
20875-60002	%2FTN	FORTRAN PASS 1	92060-13308	92001-13001	92001-13101	E
20875-60003	%3FTN	FORTRAN PASS 2	92060-13308	92001-13001	92001-13101	E
20875-60004	%4FTN	FORTRAN PASS 3	92060-13308	92001-13001	92001-13101	E
20875-60005	%5FTN	FORTRAN PASS 4	92060-13308	92001-13001	92001-13101	E
24129-60001	%ALGOL	RTE/DOS ALGOL PART 1	92060-13305	92001-13001	92001-13101	C
24129-60002	%ALGL1	RTE/DOS ALGOL PART 2	92060-13305	92001-13001	92001-13101	C
24153-60001	%FF.N	RTE/DOS FORMATTER	92060-13303	92001-13001	92001-13101	C
24170-60001	%1FTN4	RTE/DOS FORTRAN IV PART 1	92060-13306	92001-13001	92001-13101	C
24170-60002	%2FTN4	RTE/DOS FORTRAN IV PART 2	92060-13306	92001-13001	92001-13101	C
24170-60003	%3FTN4	RTE/DOS FORTRAN IV PART 3	92060-13306	92001-13001	92001-13101	C
24177-60001	%1FFT4	RTE/DOS FAST FORTRAN IV PART 1	92060-13307	92001-13001	92001-13101	1442
24177-60002	%2FFT4	RTE/DOS FAST FORTRAN IV PART 2	92060-13307	92001-13001	92001-13101	1442
24998-16001	%RLIB1	RTE/DOS LIBRARY PART 1	92060-13302	92001-13001	92001-13101	1624
24998-16001	%RLIB2	RTE/DOS LIBRARY PART 2	92060-13302	92001-13001	92001-13101	1624
24998-60002	%FF4.N	FORTRAN IV FORMATTER	92060-13303	92001-13001	92001-13101	1624
25117-60499	%DVR24	RTE 7970 7T MAG. TAPE DRIVER	92062-13301	92001-13001	92001-13101	D
29013-60001	%DVR31	RTE 7900A DISC DRIVER	92062-13301	92001-13001	92001-13101	1631
29028-60002	%DVR12	RTE 2767A DRIVER	92062-13301	92001-13001	92001-13101	A

Software Module Number	Module File Name	Module Descriptor	Part Numbers Mini-Cartridge	7900 Disc	7905 Disc	Date Code or Revision
29029-60001	%DVR00	RTE TTY/PUNCH/PHOTO READER	92062-13301	92001-13001	92001-13101	1642
29030-60001	%DVR11	RTE 2892A CARD READER DRIVER	92062-13301	92001-13001	92001-13101	B
29100-60017	!S4LP	24K SIO LINE PRINTER	92001-13305	92001-13001	92001-13101	A
29100-60018	!S4SYD	24K SIO SYSTEM DUMP	92001-13305	92001-13001	92001-13101	A
29100-60019	!S4PHR	24K SIO PHOTO READER	92001-13305	92001-13001	92001-13101	A
29100-60020	!S4PUN	24K SIO TAPE PUNCH	92001-13305	92001-13001	92001-13101	A
29100-60022	!S4L67	24K SIO 2767 LINE PRINTER	92001-13305	92001-13001	92001-13101	A
29100-60023	!S4MT2	24K SIO 7970 MAG. TAPE	92001-13305	92001-13001	92001-13101	A
29100-60049	!S4MT3	24K SIO MAG. TAPE	92001-13305	92001-13001	92001-13101	A
29100-60050	!S4TER	24K SIO TERMINAL PRINTER	92001-13305	92001-13001	92001-13101	A
59310-16002	%1DV37	RTE HP-IB WITHOUT SRQ	92062-13301	92001-13001	92001-13101	1614
59310-16003	%2DV37	RTE HP-IB WITH SRQ	92062-13301	92001-13001	92001-13101	1614
59310-16004	%HPIB	HP-IB DEVICE SUBROUTINE	92062-13301	92001-13001	92001-13101	1614
72008-60001	%1DV10	COMP. 7210A PLOTTER DRIVER	92062-13301	92001-13001	92001-13101	A
72009-60001	%2DV10	MIN. 7210A PLOTTER DRIVER	92062-13301	92001-13001	92001-13101	A
91200-16001	%DVA13	91200A DRIVER	92062-13301	92001-13301	92001-13101	1603
91200-16002	%TVLIB	91200A VIDEO MONITOR LIBRARY	92062-13301	92001-13001	92001-13101	1603
91200-16004	%TVVER	91200A TV INTERFACE VERIFIER	92062-13301	92001-13001	92001-13101	1603
92001-16002	%LDR2	RTE LOADER	92001-13301	92001-13001	92001-13101	1640
92001-16003	%MTM	MULT. TERMINAL MONITOR	92001-13301	92001-13001	92001-13101	B
92001-16004	%2DP43	POWER FAILURE DRIVER	92001-13301	92001-13001	92001-13101	1633
92001-16005	%SYLIB	RTE SYSTEM LIBRARY	92001-13301	92001-13001	92001-13101	1631
92001-16012	%CR2SY	CORE RESIDENT OPERATING SYS.	92001-13301	92001-13001	92001-13101	1642
92001-16013	!2GN00	RTE-II 7900 OFF-LINE GEN.	92001-13303	92001-13001	92001-13101	1631
92001-16014	%AUTOR	AUTO RESTART PROGRAM	92001-13302	92001-13001	92001-13101	1631
92001-16018	!2GNFH	RTE-II FIXED HEAD DISC GEN.	92001-13306	92001-13001	92001-13101	1631
92001-16020	%DVA12	2607/10/13/14/17/18 DRIVER	92062-13301	92001-13001	92001-13101	1534
92001-16026	!2GN05	RTE-II 7905 OFF-LINE GEN.	92001-13303	92001-13001	92001-13101	1631
92001-16027	%4DV05	RTE 2640A/2644A DRIVER	92062-13301	92001-13001	92001-13101	1636
92001-16028	%0DV05	RTE 2640A DRIVER	92062-13301	92001-13001	92001-13101	1636
92001-16029	%SCMD2	RTE-II COMMAND PROGRAM	92001-13301	92001-13001	92001-13101	1636
92001-16030	%WHZT2	RTE-II WHZAT PROGRAM	92001-13302	92001-13001	92001-13101	1631
92001-16031	%RT2G1	RTE-II ON-LINE GENERATOR PART 1	92001-13304	92001-13001	92001-13101	1642
92001-16031	%RT2G2	RTE-II ON-LINE GENERATOR PART 2	92001-13304	92001-13001	92001-13101	1642
92001-18014	&AUTOR	AUTO RESTART SOURCE	92001-13302	92001-13001	92001-13101	1631
92002-12001	%BMPG1	BATCH MONITOR PROGRAM PART 1	92002-13301	92001-13001	92001-13101	1631
92002-12002	%BMPG2	BATCH MONITOR PROGRAM PART 2	92002-13301	92001-13001	92001-13101	1631
92002-12001	%BMPG3	BATCH MONITOR PROGRAM PART 3	92002-13301	92001-13001	92001-13101	1631
92002-12002	%2SPO1	RTE-II SPOOL MONITOR PART 1	92002-13303	92001-13001	92001-13101	1631
92002-12002	%2SPO2	RTE-II SPOOL MONITOR PART 2	92002-13303	92001-13001	92001-13101	1631
92002-16006	%BMLIB	BATCH LIBRARY	92002-13302	92001-13001	92001-13101	1631
92002-16010	%EDITR	RTE EDITOR	92002-13302	92001-13001	92001-13101	C
92060-12004	%ASMB	RTE ASSEMBLER	92060-13304	92001-13001	92001-13101	1639
92060-16028	%XREF	CROSS REFERENCE	92060-13304	92001-13001	92001-13101	A
92060-16031	%DVR32	RTE 7905A DISC DRIVER	92062-13301	92001-13001	92001-13101	A
92060-16038	%SWTCH	RTE-II SWITCH PROGRAM	92001-13304	92001-13001	92001-13101	1636
92060-16039	%SAVE	SAVE PROGRAM	92060-13309	92001-13001	92001-13101	1642
92060-16040	%RESTR	RESTORE PROGRAM	92060-13309	92001-13001	92001-13101	1642
92060-16041	%VERFY	DISC VERIFY PROGRAM	92060-13309	92001-13001	92001-13101	1642
92060-16042	%COPY	DISC COPY PROGRAM	92060-13309	92001-13001	92001-13101	1642
92060-16043	%DBKLB	DISK BACK UP LIBRARY	92060-13309	92001-13001	92001-13101	1642
92060-16044	!DSKBK	OFF-LINE DISK BACK UP	92060-13309	92001-13001	92001-13101	1642
92060-16045	%RDNAM	READ NAMR PROGRAM	92001-13302	92001-13001	92001-13101	1631
92060-18046	&UPDAT	UPDATE TRANSFER FILE	92001-13302	92001-13001	92001-13101	1631
92060-18047	&PKDIS	PACK DISC TRANSFER FILE	92001-13302	92001-13001	92001-13101	1631
92202-16001	%DVR23	RTE 7970 9T MAG. TAPE DRIVER	92062-13301	92001-13001	92001-13101	A

MANUAL NUMBERS: 92001B

Part Number	Part Description	Manual Type
92001-93003	RTE-II PART ≠ CAT	Miscellaneous
92001-93001 92060-90012	RTE-II MANUAL RTE NEW USERS GUIDE	Software Operating System
5951-1369 5951-1374 5951-1376 5951-1390	MAN—INTRODUCTION M-SIO SYS CONFG M-BBL/BBDL/BMDL MAN—STO SUBSYS.	Software Operating Procedures

Part Number	Part Description	Manual Type
02116-9015	MNL—FORTRAN	
02116-9072	MNL—ALGOL	
24998-90001	MNL—RTE/DOS LIBR	
5951-1321	MNL—FORTRAN IV	Language and Programming
92060-90005	ASSEMBLY MANUAL	
92060-90017	RTE UTILS MNL ≠1	
92060-90020	ON-LINE GEN MNL 1	
02116-91760	M-2600 SIO PRGM	
02762-90002	M-2762/2615 SIO	
12653-90004	MNL-SIO DR	
12970-90901	MANUAL	Small Programs
12987-90006	MNL-SIO LP DRVR	
13022-90010	M-7970B SIO DR	
13029-90010	MNL-7970 7T SIO	
92060-90010	MNL POCKET GUIDE	Pocket Guide

Listed below are the software parts and manuals which are shipped when 92060B (RTE-III) is ordered (or is included in an order).

SOFTWARE MODULE NUMBERS: 92060B

| Software Module Number | Module File Name | Module Descriptor | Part Numbers | | | Date Code or Revision |
			Mini-Cartridge	7900 Disc	7905 Disc	
02607-16004	!S4L07	24K SIO LINE PRINTER DRIVER	92001-13305	92060-13001	92060-13101	1538
09601-16021	%DVR15	RTE 7261A DRIVER	92062-13301	92060-13001	92060-13101	A
12970-16004	!S4MT1	24K SIO MAG. TAPE DRIVER	92001-13305	92060-13001	92060-13101	1550
20747-60001	%DVR30	RTE FIXED HEAD DISC DRIVER	92062-13301	92060-13001	92060-13101	C
20808-60001	%CAL10	CAL. PLOTTER DRIVER	92062-13301	92060-13001	92060-13101	B
20810-60001	%CALIB	CAL. PLOTTER LIBRARY	92062-13301	92060-13001	92060-13101	C
20875-60001	%1FTN	FORTRAN MAIN CONTROL	92060-13308	92060-13001	92060-13101	E
20875-60002	%2FTN	FORTRAN PASS 1	92060-13308	92060-13001	92060-13101	E
20875-60003	%3FTN	FORTRAN PASS 2	92060-13308	92060-13001	92060-13101	E
20875-60004	%4FTN	FORTRAN PASS 3	92060-13308	92060-13001	92060-13101	E
20875-60005	%5FTN	FORTRAN PASS 4	92060-13308	92060-13001	92060-13101	E
24129-60001	%ALGOL	RTE/DOS ALGOL PART 1	92060-13305	92060-13001	92060-13101	C
24129-60002	%ALGL1	RTE/DOS ALGOL PART 2	92060-13305	92060-13001	92060-13101	C
24153-60001	%FF.N	RTE/DOS FORMATTER	92060-13303	92060-13001	92060-13101	C
24170-60001	%1FTN4	RTE/DOS FORTRAN IV PART 1	92060-13306	92060-13001	92060-13101	C
24170-60002	%2FTN4	RTE/DOS FORTRAN IV PART 2	92060-13306	92060-13001	92060-13101	C
24170-60003	%3FTN4	RTE/DOS FORTRAN IV PART 3	92060-13306	92060-13001	92060-13101	C
24177-60001	%1FFT4	RTE/DOS FAST FORTRAN IV PART 1	92060-13307	92060-13001	92060-13101	1442
24177-60002	%2FFT4	RTE/DOS FAST FORTRAN IV PART 2	92060-13307	92060-13001	92060-13101	1442
24998-16001	%RLIB1	RTE/DOS LIBRARY PART 1	92060-13302	92060-13001	92060-13101	1624
24998-16001	%RLIB2	RTE/DOS LIBRARY PART 2	92060-13302	92060-13001	92060-13101	1624
24998-16002	%FF4.N	FORTRAN IV FORMATTER	92060-13303	92060-13001	92060-13101	1624
25117-60499	%DVR24	RTE 7970 7T MAG. TAPE DRIVER	92062-13301	92060-13001	92060-13101	D
29013-60001	%DVR31	RTE 7900A DISC DRIVER	92062-13301	92060-13001	92060-13101	1631
29028-60002	%DVR12	RTE 2767A DRIVER	92062-13301	92060-13001	92060-13101	A
29029-60001	%DVR00	RTE TTY/PUNCH/PHOTO READER	92062-13301	92060-13001	92060-13101	1642
29030-60001	%DVR11	RTE 2892A CARD READER DRIVER	92062-13301	29060-13001	92060-13101	B
29100-60017	!S4LP	24K SIO LINE PRINTER	92001-13305	92060-13001	92060-13101	A
29100-60018	!S4SYD	24K SIO SYSTEM DUMP	92001-13305	92060-13001	92060-13101	A
29100-60019	!S4PHR	24K SIO PHOTO READER	92001-13305	92060-13001	92060-13101	A
29100-60020	!S4PUN	24K SIO TAPE PUNCH	92001-13305	92060-13001	92060-13101	A
29100-60022	!S4L67	24K SIO 2767 LINE PRINTER	92001-13005	92060-13001	92060-13101	A
29100-60023	!S4MT2	24K SIO 7970 MAG. TAPE	92001-13305	92060-13001	92060-13101	A
29100-60049	!S4MT3	24K SIO MAG. TAPE	92001-13305	9206 13001	92060-13101	A
29100-60050	!S4TER	24K SIO TERMINAL PRINTER	92001-13305	92060-13001	92060-13101	A
59310-16002	%1DV37	RTE HP-IB WITHOUT SRQ	92062-13301	92060-13001	92060-13101	1614
59310-16003	%2DV37	RTE HP-IB WITH SRQ	92062-13301	92060-13001	92060-13101	1614
59310-16004	%HPIB	HP-IB DEVICE SUBROUTINE	92062-13301	92060-13001	92060-13101	1614
72008-60001	%1DV10	COMP. 7210A PLOTTER DRIVER	92062-13301	92060-13001	92060-13101	A

about the HP 1000

about the HP 1000

Software Module Number	Module File Name	Module Descriptor	Part Numbers Mini-Cartridge	Part Numbers 7900 Disc	Part Numbers 7905 Disc	Date Code or Revision
72009-60001	%2DV10	MIN. 7210A PLOTTER DRIVER	92062-13301	92060-13001	92060-13101	A
91200-16001	%DVA13	91200A DRIVER	92062-13301	92060-13001	92060-13101	1603
91200-16002	%TVLIB	91200A VIDEO MONITOR LIBRARY	92062-13301	92060-13001	92060-13101	1603
91200-16004	%TVVER	91200A TV INTERFACE VERIFIER	92062-13301	92060-13001	92060-13101	1603
92001-16003	%MTM	MULT. TERMINAL MONITOR	92060-13301	92060-13001	92060-13101	B
92001-16005	%SYLIB	RTE SYSTEM LIBRARY	92060-13301	92060-13001	92060-13101	1631
92001-16014	%AUTOR	AUTO RESTART PROGRAM	92060-13310	92060-13001	92060-13101	1631
92001-16020	%DVA12	2607/10/13/14/17/18 DRIVER	92062-13301	92060-13001	92060-13101	1534
92001-16027	%4DV05	RTE 2640A/2644A DRIVER	92062-13301	92060-13001	92060-13101	1636
92001-16028	%0DV05	RTE 2640A DRIVER	92062-13301	92060-13001	92060-13101	1636
92001-18014	&AUTOR	AUTO RESTART PROGRAM SOURCE	92060-13310	92060-13001	92060-13101	1631
92002-12001	%BMPG1	BATCH MONITOR PROGRAM PART 1	92002-13301	92060-13001	92060-13101	1631
92002-12001	%BMPG2	BATCH MONITOR PROGRAM PART 2	92002-13301	92060-13001	92060-13101	1631
92002-12001	%BMPG3	BATCH MONITOR PROGRAM PART 3	92002-13301	92060-13001	92060-13101	1631
92002-16006	%BMLIB	BATCH LIBRARY	92002-13302	92060-13001	92060-13101	1631
92002-16010	%EDITR	RTE EDITOR	92002-13302	92060-13001	92060-13101	C
92060-12001	%3SPO1	RTE-III SPOOL MONITOR PART 1	92060-13313	92060-13001	92060-13101	1631
92060-12001	%3SPO2	RTE-III SPOOL MONITOR PART 2	92060-13313	92060-13001	92060-13101	1631
92060-12003	%CR3SY	MEMORY RESIDENT SYSTEM	92060-13301	92060-13001	92060-13101	1642
92060-12004	%ASMB	RTE ASSEMBLER	92060-13304	92060-13001	92060-13101	1639
92060-16001	%3DP43	POWER FAILURE DRIVER	92060-13301	92060-13001	92060-13101	1633
92060-16004	%LDR3	RTE-III LOADER	92060-13301	92060-13001	92060-13101	1640
92060-16006	%WHZT3	RTE-III WHZAT PROGRAM	92060-13310	92060-13001	92060-13101	1631
92060-16028	%XREF	CROSS REFERENCE	92060-13304	92060-13001	92060-13101	A
92060-16029	!3GN00	7900 RTE-III GENERATOR	92060-13311	92060-13001	92060-13101	1631
92060-16031	%DVR32	RTE 7905A DISC DRIVER	92062-13301	92060-13001	92060-13101	A
92060-16032	!3GN05	7905 RTE-III GENERATOR	92060-13311	92060-13001	92060-13101	1631
92060-16035	%$PVMP	SPVMP	92060-13301	92060-13001	92060-13101	A
92060-16036	%SCMD3	RTE-III COMMAND PROGRAM	92060-13301	92060-13001	92060-13101	1636
92060-16037	%RT3G1	RTE-III ON-LINE GENERATOR PART 1	92060-13312	92060-13001	92060-13101	1642
92060-16037	%RT3G2	RTE-III ON-LINE GENERATOR PART 2	92060-13312	92060-13001	92060-13101	1642
92060-16038	%SWTCH	RTE-III SWITCH PROGRAM	92060-13312	92060-13001	92060-13101	1636
92060-16039	%SAVE	SAVE PROGRAM	92060-13309	92060-13001	92060-13101	1642
92060-16040	%RESTR	RESTORE PROGRAM	92060-13309	92060-13001	92060-13101	1642
92060-16041	%VERFY	DISC VERIFY PROGRAM	92060-13309	92060-13001	92060-13101	1642
92060-16042	%COPY	DISC COPY PROGRAM	92060-13309	92060-13001	92060-13101	1642
92060-16043	%DBKLB	DISK BACK UP LIBRARY	92060-13309	92060-13001	92060-13101	1642
92060-16044	!DSKBK	OFF LINE DISK BACK UP	92060-13309	92060-13001	92060-13101	1642
92060-16045	%RDNAM	READ NAMR PROGRAM	92060-13310	92060-13001	92060-13101	1631
92060-18046	&UPDAT	UPDATE TRANSFER FILE	92060-13310	92060-13001	92060-13101	1631
92060-18047	&PKDIS	PACK DISK TRANSFER FILE	92060-13310	92060-13001	92060-13101	1631
92202-16001	%DVR23	RTE 7470 9T MAG. TAPE DRIVER	92062-13301	92060-13001	92060-13101	A

MANUAL NUMBERS: 92060B

Part Number	Part Description	Manual Type
92060-90004	RTE-III MANUAL	Software Operating System
92060-90012	RTE NEW USERS GUIDE	
5951-1369	MAN—INTRODUCTION	
5951-1374	M-SIO SYS CONFG	Software Operating Procedures
5951-1376	M-BBL/BBDL/BMDL	
5951-1390	MAN—SIO SUBSYS.	
02116-9015	MNL—FORTRAN	
02116-9072	MNL—ALGOL	
24998-90001	MNL—RTE/DOS LIBR	
5951-1321	MNL—FORTRAN IV	Language and Programming
92060-90005	ASSEMBLY MANUAL	
92060-90017	RTE UTILS MNL ≠1	
92060-90020	ON-LINE GEN MNL 1	
02116-91760	M-2600 SIO PRGM	
02762-90002	M-2762/2615 SIO	
12653-90004	MNL—SIO DR	
12970-90901	MANUAL	Small Programs
12987-90006	MNL—SIO LP DRVR	

Part Number	Part Description	Manual Type
13022-90010 13029-90010	M-7970B SIO DR MNL—7970 7T SIO	Small Programs Continued
92060-90019	RTE III PT # CAT	Miscellaneous
92060-90010	MNL POCKET GUIDE	Pocket Guide

RTE DRIVERS

Following is a list of the drivers available for RTE systems.

RTE DRIVERS

Software Module Number	Module File Name	Module Descriptor	Part Numbers			Date Code or Revision
			Mini-Cartridge	7900 Disc	7905 Disc	
09601-16021	%DVR15	RTE 7261A DRIVER	92062-13301			A
20747-60001	%DVR30	RTE FIXED HEAD DISC DRIVER	92062-13301			C
20808-60001	%CAL10	CAL. PLOTTER DRIVER	92062-13301			B
20810-60001	%CALIB	CAL. PLOTTER LIBRARY	92062-13301			C
25117-60499	%DVR24	RTE 7970 7T MAG. TAPE DRIVER	92062-13301			D
29013-60001	%DVR31	RTE 7900A DISC DRIVER	92062-13301			1631
29028-60002	%DVR12	RTE 2767A DRIVER	92062-13301			A
29029-60001	%DVR00	RTE TTY/PUNCH/PHOTO READER	92062-13301			1642
29030-60001	%DVR11	RTE 2892A CARD READER DRIVER	92062-13301			B
59310-16002	%1DV37	RTE HP-IB WITHOUT SRQ	92062-13301			1614
59310-16003	%2DV37	RTE HP-IB WITH SRQ	92062-13301			1614
59310-16004	%HPIB	HP-IB DEVICE SUBROUTINE	92062-13301			1614
72008-60001	%1DV10	COMP. 7210A PLOTTER DRIVER	92062-13301			A
72009-60001	%2DV10	MIN. 7210A PLOTTER DRIVER	92062-13301			A
91200-16001	%DVA13	91200A DRIVER	92062-13301			1603
91200-16002	%TVLIB	91200A VIDEO MONITOR LIBRARY	92062-13301			1603
91200-16004	%TVVER	91200A TV INTERFACE VERIFIER	92062-13301			1603
92001-16020	%DVA12	2607/10/13/14/17/18 DRIVER	92062-13301			1534
92001-16027	%4DV05	RTE 2640A/2644A DRIVER	92062-13301			1636
92001-16028	%0DV05	RTE 2640A DRIVER	92062-13301			1636
92060-16031	%DVR32	RTE 7905A DISC DRIVER	92062-13301			A
92202-16001	%DVR23	RTE 7470 9T MAG. TAPE DRIVER	92062-13301			A

DOS-IIIB MODULES

The Index below indicates the modules available for DOS-IIIB systems, HP 24307B, date code 1523.

This Index relates the names of the relocatable modules to the part numbers of the equivalent paper tapes and indicates the purpose of the modules. Modules not specifically designated for the 2100A/S or for the 21MX computers are to be used on either.

NAME	PART NUMBER	REV	DESCRIPTION
DISCH	24307-16069	1523	DISC MONITOR
SEXMD	24307-16070	1523	EXEC MODULES
DVR00	20985-60001	1516	TTY-LIKE CONSOLE/ TERMINAL
DVR01	20987-60001	1419	PAPER TAPE READER
DVR02	20989-60001	1419	PAPER TAPE PUNCH

DVR05	24157-60001	1419	TTY-LIKE CONSOLE
DVR15	24307-16017	1446	7261A MARK SENSE CARD READER
D2892	24272-60001	1419	2892B CARD READER (DVR11)
D2767	24168-60001	1419	2767A LINE PRINTER (DVR12)
D26XX	24307-16011	1446	DVR12 FOR 2607, 2610, 2614, 2613, 2618
DVR23	13024-60001	1446	7970B/E MAG TAPE
DVR26	24307-16018	1507	2762A/B AND 2615A CONSOLE
DVR30	24307-16073	1523	DISC BATCH DRIVER
DVR31	24156-60001	1419	7900/7901 DISC
DVR67	24341-60001	1419	12889A HI SPD SERIAL IF
DVR70	24307-16009	1446	DVR70 FOR 12618A SYNC INTERFACE
DVR71	24307-16013	1515	12967A SYNC MODEM IF
DVR72	24350-16001	1523	12587B ASYNC DATA SET IF

about the HP 1000

NAME	PART NUMBER	REV	DESCRIPTION
DVR73	24377-16001	1523	12920A/B MULTIPLEXOR
DVR74	24307-16014	1515	12966A/12968A ASYNCH IF
EFMP	24309-60002	1523	EXT FILE MGR EXEC MODULES
	24309-60003	1523	EXT FILE MGR UTILITIES
JOBPR	24307-16071	1523	JOB PROCESSOR
RLODR	24307-16072	1523	RELOCATING/LINKING LOADER
ASMB	24307-16006	1419	2100/21MX ASSEMBLER
.FTN4	24170-60001	C	FORTRAN IV COMPILER
	24170-60002	C	
	24170-60003	C	
FTN4	24177-60001	1442	FORTRAN IV COMPILER (10K AREA)
	-60002	1442	
ALGOL	24129-60001	C	ALGOL COMPILER
	24129-60002	C	
XREF	24223-60001	1523	2100/21MX CROSS REF TABLE GEN
F4D.N	24152-60001	C	RELO SUBRLIBR FTN4
F2E.N	24151-60001	D	RELO SUBR LIBR (EAU)
F2F.N	24248-60001	B	RELO SUBR LIBR (FP)
FFP.N	12907-16001	A	2100A/S FFP SUBR LIBRARY
$SETP	12907-16002	1350	2100A/S FFP SUBR $SETP
ATD01	24381-16001	1503	ASYNC TERMINAL DRIVER No. 1
ATD02	24307-16012	1442	ASYNC TERMINAL DRIVER No. 2
PMT01	24307-16008	1438	PAGE MODE TERMINAL DRIVER No. 1
PMT02	24307-16016	1503	PAGE MODE TERMINAL DRIVER No. 2
ACR01	07260-16001		ASYNC CARD READER DRIVER No. 1 for 7260A
SLC	24307-16010	1438	SYNCHRONOUS LINE CONTROL DRIVER
DVR33	24278-60001	1419	2100/21MX WCS DRIVER
MASMB	24332-60001	1419	2100A/S WCS MICRO ASSEMBLER
WCSUT	24333-60001	A	2100/21MX MICRO UTILITIES
MDBUG	24334-60001	1419	2100A/S WCS MICRO DEBUG EDITR
XASMB	12978-16001	1437	21MX WCS MICRO ASSEMBLER
XDBUG	12978-16002	1437	21MX WCS MICRO DEBUG EDITOR
FFP.X	12977-16001	1451	21MX FFP SUBR LIBRARY
XSETP	12977-16002	1451	21MX FFP SUBR $SETP

documentation

The following tables list currently available customer manuals for Data Systems Division products. This list supersedes the list in the last issue of the **Communicator.**

The most recent changes to the tables are indicated for easy reference. Prices are subject to change without notice.

Copies of manuals and updates can be obtained from your local Sales and Service office. The address and telephone number of the office nearest to you are listed in the back of all customer manuals.

Update packages are free of charge. If you require an update package only, send your request to:

Software/Publications Distribution
11000 Wolfe Road
Cupertino, Ca. 95014

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manual(s) you need on the Corporate Parts Center form supplied at the back of the **Communicator**.

A few words about documentation terms:

New   A new manual refers only to the first printing of a manual. When first printed, a manual is assigned a part number.

Revised   A revised manual is a printing of an existing manual which incorporates new and/or changed information in its contents. For example, a manual is revised when an update package is incorporated into the manual: the manual gets a new print date and the update package disappears. Note that a revision to a manual effectively obsoletes the previous version of the manual.

Update   An update package is a supplement to an existing manual which contains new and/or changed information. Updates are issued when information must get to customers, yet it is inappropriate to issue a revised manual. An update has no part number; it is automatically included when you order the manual with which it is associated.

1000 SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02172-90002	2172A Computer System	$ 3.00	9/76*N	
02172-90005	Getting Started with Your HP1000	1.50	9/76*N	
91780-93001	RJE/1000 Programming Manual	9.50	11/76*N	

RTE SYSTEMS MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02170-90001	Site Preparation, Installation and Service Manual	$ 3.00	11/76*N	
02171-90001	Site Preparation, Installation and Service Manual	3.00	11/76*N	
02313-93002	RTE 2313B Analog-Digital Interface Subsystem Operating and Service Manual	30.00	8/76	
02320-93002	RTE System Driver DVR76 for HP 2320A Low Speed Data Acquisition Subsystem Programming and Operating Manual	1.00	8/74	
02321-93001	RTE System Driver DVR74 for HP 2321A Low Speed Data Acquisition Subsystem Programming and Operating Manual	1.00	8/74	
09600-93010	RTE System DVR11 for HP 2892A Card Reader Programming and Operating Manual	1.00	8/74	
09600-93015	91200B TV Interface Kit; Programming and Operating Manual	4.50	7/75	1/76
09601-93007	RTE Device Subroutine for HP 5327A/B-H48 Counter	2.50	12/74	
09601-93009	RTE Device Subroutine for HP 5326A-H18 Counter	2.50	12/74	
09601-93015	RTE for 40-bit Output Register #12556B	1.00	10/74	
09603-93001	9603A/9604A Control System and Scientific Measurement Operating and Service Manual	7.50	5/76	
09610-93003	ISA FORTRAN Extension Package Reference Manual	4.50	7/76	
09611-90009	9611A Operating 406 Industrial Measurement and Control System	.25	4/75	
09611-90010	HP 6940A/B Multiprogrammer Verification Manual	4.50	8/75	
12604-93002	RTE DVR40 for 12604B Data Source Interface	1.00	8/74	
12665-93001	RTE System Driver DVR65 for HP 12771A Computer Serial Interface Kit	1.00	8/74	
12989-99001	RTE System Driver DVA15 for Card Reader Punch Subsystem 2894	1.00	1/75	5/76
13197-90001	RTE Driver DVR36 Programming and Operating Manual	3.00	9/76*N	
24998-90001	DOS/RTE Relocatable Library Reference Manual	10.00	3/76	
25117-93003	RTE System Driver DVR24 for HP 7970 Series Digital Magnetic Tape Unit	1.00	8/74	
29003-93001	RTE System Driver DVR66 for HP 12772A Coupler Modem Interface Kit Programming and Operating Manual	1.00	8/74	
29003-93003	RTE System Driver DVR66 for HP 12770A Coupler Serial Interface Kit Programming and Operating Manual	1.00	8/74	
29009-93001	RTE System Driver DVR62 for HP 2313B Subsystem	2.50	8/74	
29013-90001	DVR31 RTE Moving Head Driver	10.00	2/73	
29015-90001	Fixed Head Real-Time System Generator	15.00	4/72	
29016-90002	RTE Scheduler	50.00	9/72	
29016-90003	Read-Time Input/Output Control	50.00	12/73	

*O = Obsolete Manual   *R = Revised Manual   *N = New Manual

about the HP 1000

about the HP 1000

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
29028-95001	RTE HP 2610A/2614A Line Printer Driver	1.50	8/73	
29029-91001	Real-Time Executive Multiple-Device System Control Device (DVR00) Program Listing	10.00	9/72	
29029-95001	Real-Time Executive System Driver DVR00 for Multiple Device System Control Small Programs Manual	1.50	11/75	
29100-93001	RTE System Driver DVR40 (29100-60041) for HP 12604B Data Source Interface Programming and Operating Manual	1.00	8/76	
29110-93003	RTE System Driver DVR61 for HP 6940A, 6941A Bidirectional Multiprogrammer Programming and Operating Manual	4.50	3/76	
29101-93001	RTE Core-Based Software System Users Manual	$10.00	1/76	
29102-93001	RTE BASIC Software System Programming and Operating Manual	10.00	3/74	8/75
29103-93001	RTE System Cross Loader; Programming and Operating Manual	2.50	3/75	11/75
91060-93005	RTE Driver for X-Y Display Storage Subsystem (HP Model 1331C-016) Programming and Operating Manual	1.00	8/74	
91062-93003	Real-Time Executive System Driver for DVM/Scanner Subsystem	9.00	8/74	
91700-93001	Distribured System CCE Operating Manual	20.00	2/76	5/76
91705-93001	Distributed System SCE/5 Operating Manual	15.00	7/75	2/76
92001-90015	RTE DVR05 for 264X Terminals	2.00	9/76	
92001-93001	RTE-II Software System Programming and Operating Manual	10.00	7/76	5/76
92060-90004	RTE-III Software System Programming and Operating Manual	12.00	7/76	
92060-90005	RTE Assembler Reference Manual	7.00	1/76	
92060-90009	RTE-III General Information Manual	4.00	2/76	
92060-90010	RTE Batch/Spool Monitor and Operating System Pocket Guide	3.00	10/75	
92060-90012	RTE: A Guide for New Users	6.50	7/76	
92060-90013	Batch-Spool Monitor Reference Manual	9.50	7/76	
92060-90014	RTE Interactive Editor Reference Manual	6.00	3/76	
92060-90016	Multi-User Real-Time BASIC Reference Manual	12.00	10/75	12/75
92060-90017	RTE Utility Programs	3.00	7/76	
92060-90020	RTE On-Line Generator	15.00	7/76	9/76
92063-90001	IMAGE/1000 Data Base Management System Reference Manual	9.00	11/76	
92200-93001	RTE System Driver DVR12 for HP 2607A Line Printer Programming and Operating Manual	1.00	3/74	
92200-93005	Real-Time Executive Operating System Drivers and Device Subroutine Manual	5.00	7/76	
92202-93001	RTE System Driver DVR23 for HP 7970 Series Digital Mag Tape Units Programming and Operating Manual	1.00	8/74	
92400-93001	92400A Utility Library Subroutine for Sensor-Based Diagonistics	7.50	11/76*N	
93005-93005	Thermal Line Printer Subsystem for Driver DVR00 (RTE)	2.50	12/74	
93513-90002	RTE System Driver DVA76-DVR40 for 2801 Quartz Thermometer System	1.50	4/75	

SOFTWARE NUMBERING CATALOG MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
92060-90019	92060B Software Numbering Catalog	$ 1.50	7/76	
92001-93003	92001B Software Numbering Catalog	1.50	7/76	
92002-93003	92002A Software Numbering Catalog	1.50	7/76	
91705-93003	91705A Software Numbering Catalog	1.00	7/76	
92413-90001	92413A Software Numbering Catalog	1.00	7/76	

*N = New Manual

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
91780-93004	91780A Software Numbering Catalog	1.00	7/76	
92062-90001	92062A Software Numbering Catalog	1.00	7/76	
92101-90001	92101A Software Numbering Catalog	1.00	7/76	
92409-93002	92409A Software Numbering Catalog	1.00	7/76	
92400-93003	92400A Software Numbering Catalog	1.00	7/76	
91703-93003	91703A Software Numbering Catalog	1.00	7/76	
91704-93003	91704A Software Numbering Catalog	1.00	7/76	
91700-93005	91700A Software Numbering Catalog	1.00	7/76	
92063-90003	92063A Software Numbering Catalog	1.00	9/76	
92066-90001	92066A Software Numbering Catalog	1.00	9/76	

SOFTWARE INPUT/OUTPUT SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02116-91760	Teleprinter Driver (LP Compatible) Manual	$ 1.00	8/73	
02762-90002	HP 2762A Terminal Printer Driver	1.00	5/73	
02892-90003	HP 2892A Card Reader Driver	1.50	6/72	
12602-90022	Mark Sense Card Reader Drivers	1.00	6/70	
12653-90004	HP 2767 Line Printer Driver	1.00	9/70	1/73
12845-90005	HP 2610A/2614A Line Printer Driver	1.00	2/74	
12987-90006	HP 2607 Line Printer Driver	5.00	11/73	
13022-90010	HP 7970 Magnetic Tape Unit Driver	1.00	2/72	
13029-90010	Magnetic Tape Driver (7-Track)	1.00	2/72	
5950-9276	SIO Drum-Disc	1.00	2/70	
5951-1374	Software Input/Output System Configuration	1.00	7/74	.
5951-1390	Subsystem Operation	2.00	2/76	
59310-90063	RTE DVR37 459310 B Interface Bus Programming and Operating Manual	3.50	5/76	

BASIC CONTROL SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02022-90014	Magnetic Tape Reformatting System Support Utilities	$ 1.50	1/74	
02100-90129	HP 2100 Microassembler Coding Form	3.00		
02100-90140	Decimal String Arithmetic Routines	5.00	10/73	
02100-90200	Loader/Loader Reference Manual	2.00	5/76	
02108-90008	Microprogramming 21MX Computers Reference Manual	6.50	2/76	5/76
02116-9017	Basic Control System Manual	8.50	12/71	
02116-9072	ALGOL for HP 2000 Computers Reference Manual	10.00	2/76	
02116-91751	Prepare Tape System	2.50	8/74	
02116-91752	Magnetic Tape System	6.00	6/71	
02116-91780	2100 Series Relocatable Subroutines	11.00	12/74	
02762-90003	HP 2762A Terminal Printer Driver	1.00	5/73	
02892-90004	HP 2892A Card Reader Driver	1.50	6/72	
12602-90021	Mark Sense Drivers	1.00	6/70	

*N = New Manual

about the HP 1000

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
12653-90005	HP 2767 Line Printer Driver	1.00	10/70	
12845-90004	HP 2610A/2614A Line Printer Driver	1.00	6/72	
12987-90008	HP 2607 Line Printer Driver	5.00	12/73	
13023-90010	HP 7970 Magnetic Tape Unit Driver	1.00	5/74	
13026-90010	Magnetic Tape Driver (7 Track without DMA)	1.00	5/71	6/72
13027-90010	Magnetic Tape Driver (7-Track with DMA)	1.00	5/71	6/72
5951-1371	HP 2100 Front Panel Procedures	1.00	8/73	
5951-1376	Basic Binary Loader/Disc Loader, Basic Moving-Head Disc Loader	2.50	4/76	
5951-1391	Basic Control System	1.50	10/74	
5951-1392	Magnetic Tape System	1.00	7/71	

DISC OPERATING SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02767-00007	DOS/RTE 2767 Line Printer Driver	$ 1.00	12/70	
12560-90023	DOS RTE and BCS Calcomp Plotter Drivers	1.50	10/75	
12602-90023	DOS/RTE Mark Sense Drivers Kit 12602B	1.00	8/70	
12908-90004	HP 12908 Writable Control Store Driver	1.00	2/75	
24307-90006	DOS-III Reference Manual	20.00	1/76	3/76
24307-90012	DOS-III Data Communications Drivers	7.50	4/76	
24307-90018	DOS-III Pocket Guide	3.50	12/75	
24307-90022	DOS-III Terminal Printer Driver	1.00	1/75	
24307-90073	DOS-III Standard Drivers	6.00	1/75	
24376-90001	IMAGE/2000 Data Base Management System Reference Manual	11.00	8/75	
5951-1366	Cross Reference Table Generator	1.00	8/74	

LANGUAGE MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02116-9014	HP Assembler Manual	$ 6.50	8/75	
02116-9015	HP FORTRAN Manual	5.00	3/74	
02116-9016	Symbolic Editor	4.50	2/74	
02116-9072	ALGOL Reference Manual	10.00	2/76	
12907-90010	Implementing the HP 2100 Fast FORTRAN Processor	1.00	7/76	
24307-90014	DOS-III Assembler Reference Manual	8.00	7/74	11/75
92060-90005	RTE Assembler Reference Manual	7.00	1/76	
5951-1321	HP FORTRAN IV Reference Manual	6.00	12/75	

training schedule

The schedule for customer training courses on Data Systems Division products has been expanded to include courses offered at our European training centers. Listed below are courses offered in the U.S. and in Europe during the period January, 1977 through April, 1977.

You can also obtain a copy of the training schedule from your local HP sales office. A European course schedule is available through the sales offices in Europe; a U.S. schedule through U.S. sales offices.

*Prices quoted are for courses at the two U.S. training centers only. For prices of courses at European training centers please consult your local HP Sales Office.

Registration

Requests for enrollment in any of the above courses should be made through your local HP representative. He will supply the Training Registrar at the appropriate location with the course number, dates, and requested motel reservations. Enrollments are acknowledged by a written confirmation indicating the Training Course, time of class, location and accommodations reserved.

Accommodations

Students provide their own transportation, meals and lodging. The Training Registrar will be pleased to assist in securing motel reservations at the time of registration.

Cancellations

In the event you are unable to attend a class for which you are registered please notify the Training Center Registrar immediately in order that we may offer your seat to another student.

Training Center Addresses

Cupertino

11000 Wolfe Road
Cupertino, California 95014
(408) 257-7000

Sunnyvale

974 East Arques
Sunnyvale, California

Rockville

4 Choke Cherry Road
Rockville, Maryland 20850
(301) 948-6370

Boise

P.O. Box 15
15 N. Phillippi Street
Boise, Idaho 83707
(208) 376-6000
TWX: 910-970-5784

Boblingen

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

Winnersh

King Street Lane
GB-Winnersh, Wokingham
Berks RG11 5 AR.
Tel: Wokingham 784774
Cable: Hewpie London
Telex: 847178 9

Grenoble

5, avenue Raymond-Chanas
38320 Eybens
Tel: (76) 25-81-41
Telex: 980124

Milan

Via Amerigo Vespucci, 2
1-20124 Milan
Tel: (2) 62 51
Cable: HEWPACKIT Milano
Telex: 32046

Madrid

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

Stockholm

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

about the HP 1000

TITLE — TRAINING COURSE RATES AND CENTER LOCATION

Course Number	Length	Price	Cupertino	Sunnyvale	Rockville	Boise	Boblingen	Winnersh	Grenoble	Milan	Madrid	Stockholm	Amsterdam/ Brus.
22940A	2100 Maint. 10 days	$1000		Jan 17, Mar 14, Apr 11									
22941A	21MX Maint. 5 days	500		Jan 3, Jan 17, Feb 7, Feb 28					Jan 31, May 9				
22942A	7900 Maint. 5 days	500		Jan 10, Mar 28					Mar 7				
22943A	7970B Maint. 5 days	600				Mar 7, Apr 25			Jan 24, Apr 4				
22944A	7970E Maint. 5 days	600				Jan 31, Apr 18							
22945A	7905 Maint. 5 days	500		Jan 31, Mar 7, Apr 25					Feb 21, Apr 25				
22950A	2100 Ser. Assm. 5 days	500	Jan 3, Jan 31, Feb 28, Mar 21, Apr 11		Jan 17, Jan 31, Mar 14, Apr 25		Feb 7, Mar 28, May 23	Feb 7, Apr 18	Feb 21, May 2, Jun 13	Jan 10, May 30	Feb 7, May 2	Feb 21, Apr 18, May 23	Mar 7, Jun 6
22952A	DOS III B 5 days	500	**				Jan 17, May 9						
22953A	2100 Image 3 days	300											
22959A	Assembler/21MX 5 days	500	Jan 3, Jan 31, Feb 28, Mar 21, Apr 11					Feb 7, Apr 18, Jun 13, Aug 8					
22960A	21MX Mic. Prog. 5 days	500	Feb 7, Mar 28						Feb 14				
22965B	RTE-II/III 10 days (Course includes RTE-II/III operating system, batch spool monitor and file manager.)	1000	*{Jan 17/Jan 24}, *{Jan 31/Feb 7}, *{Feb 7/Feb 14}, *{Feb 28/Mar 7}, *{Mar 7/Mar 14}, *{Mar 21/Mar 28}, *{Apr 11/Apr 18}, *{Apr 18/Apr 25}		Jan 24, Feb 7, Feb 28, Mar 21, Apr 11		*{Jan 24/Jan 31}, *{Feb 21/Feb 28}, *{Mar 21/Mar 28}, *{Apr 25/May 2}	*{Jan 10/Jan 17}, *{Mar 14/Mar 21}, *{May 9/May 16}, {Jul 11/Jul 18}	*{Jan 17/Jan 31}, *{Feb 28/Mar 14}, *{Apr 4/Apr 18}, *{May 9/May 23}	*{Jan 24/Feb 7}, *{Mar 7/Mar 21}	*{Feb 14/Feb 21}, {May 9/May 16}	*{Feb 28/Mar 7}, *{Apr 25/May 2}	*{Jan 10/Jan 24}, *{Mar 28/Apr 18}
22968A	Measurement & control 2 days	200					Mar 10		Jan 24, Jun 27				
22969A	Distb. Sys. 5 days	500	Jan 17, Feb 14, Mar 28		Mar 14		Jan 10, Apr 18		Mar 7				May 9
22977A	Image/DBMS 1000 5 days	500	Jan 24, Mar 21, Apr 25		Jan 17, Apr 25		May 9		Feb 7, Apr 25	Apr 4	Feb 28, May 23		Feb 14
22978	TCS 2 days	200	*										

TITLE   **TRAINING COURSE RATES AND CENTER LOCATION**

Course Number	Length	Price	Cupertino	Sunnyvale	Rockville	Boise	Boblingen	Winnersh	Grenoble	Milan	Madrid	Stockholm	Amsterdam/ Brus.
22979A	Real/Time Multiterminal Basic		Feb 23 Apr 4		**		Mar 7						
	3 days	300											
22980A	HPIB Multicomputer Bus Basic		Jan 24 Mar 14 Apr 25						Mar 21				
	3 days	300											
22981A	HPIB Programming Under RTE		Jan 27 Mar 17 Apr 28						Feb 14 Mar 24				
	2 days	200											

*NOTE: Dates within brackets are starting dates for week 1 and week 2 of the RTE course. In some cases there is a break between the two weeks of the class. Course 22977A, IMAGE/DBMS 1000 replaces 22953A (2100 IMAGE); the new class adds additional material and extends the training from 3 to 5 days.

**On Sufficient Demand.

(vertical text, right margin) about the HP 1000

about the HP 2000

software tips

LINE PRINTER SELECTION FOR HP 2000 REMOTE JOB ENTRY

Terry Eastham
HP General Systems

The multileaving protocol used with HP 2000 HASP RJE automatically gives the user "trailing blank" and "repeat character" compression. This means, for example, that a 132 character line from a page consisting of two ten-character columns of information would transmit as only twenty characters plus some control characters. The benefit to the user is that depending on the printing mix, a faster line printer may be driven very effectively.

At the same time, line printer throughput may be limited by the modem transmission speed. For example, a 4800 bps modem can transfer data at 600 characters per second which is equivalent to about 270 uncompressed print lines per minute. Thus, given a slow modem and low compressibility, a faster line printer will not result in faster printing.

The table below shows maximum possible line printer throughputs for several printing mixes and modem speeds. Remember that if your RJE line is noisy or the host computer is slow refilling your buffers the actual throughput could be much less.

	2400* (HD)	2400 (FD)	4800* (HD)	4800 (FD)	9600 (FD)
5. HASP Job ID Page; unlikely to occur for more than 1 or 2 lines (blank lines or lines up to 132 identical characters)	2260	2400	4240	4640	8730

NOTE: HD = Half Duplex   FD = Full Duplex
        CPL = Characters Per Line

*Half Duplex operation assumes 50 millisecond "line turn-around time".

bulletins

OPTICAL MARK READERS NOW SUPPORTED ON HP 2000 AND HP 3000 SERIES II SYSTEM

Bernard Guidon
HP Boise Division

The 7260A Optical Mark Reader is now supported on both HP 2000 and HP 3000 computer systems, offering mark sense document reading capability on system terminals. Adding 7260A's on either system with an existing terminal does not require any hardware or software modifications. It is done simply by inserting the 7260A between the terminal and the computer. The advantages of this combination include:

MAXIMUM PRINT LINES PER MINUTE (approx.)

	2400* (HD)	2400 (FD)	4800* (HD)	4800 (FD)	9600 (FD)
1. Text, Accounting Expense Reports (132 CPL, no more than 2 consecutive identical characters, including blanks)	110	120	210	230	430
2. Typical file listings (62 non-blank CPL, arranged in 11 columns)	190	200	350	390	730
3. Typical MK IV or COBOL compiler output (40 CPL arranged in 7 columns)	280	290	520	570	1070
4. Very simple MK IV report (20 CPL arranged in 3 columns)	600	620	1120	1210	2300

- Saves data/program preparation time and prevents errors by using one functional card for both source document and data entry.

- The data may be marked with an ordinary pencil.

- Data input transmission can be performed at maximum OMR speed.

- Much faster than manual keying.

- Easy to control through use of BASIC language PRINT statements on 2000 systems and through use of the FCARD intrinsic on 3000 systems.

- ASCII or IMAGE reading modes.

- Switchable OFF LINE operation with terminals.

2000 SYSTEM

Full control of the OMR is provided to the user through use of BASIC PRINT statements. The 7260A with 2000 System Application Brief (HP P/N 5953-0101), available from any HP sales office, fully describes system installation and operation of the OMR.

Greater capability is also offered by using the LOAD command of the 2000 system. This permits a terminal user to request that a specified disc file is loaded into his work space as a program. Assuming the specified file contains ASCII data in the format of a BASIC program, the user is then able to run this program like another. During the loading any errors of syntax, etc., are output on the user's terminal and may be subsequently corrected.

This powerful capability permits one to use educational BASIC cards (HP P/N 9320-2051) through 7260 Optical Mark Reader terminals connected to a 2000 System. The cards have been layed out in a special easy-to-mark format (not Hollerith) so that even young school children can use them. A small BASIC program is all that is required to interpret the marks and build a file of BASIC language statements.

This new capability made possible by the LOAD command dramatically reduces the cost of providing student computing retaining a fast turn around by using terminal reader combinations located in each school, college and study center.

3000 SERIES II SYSTEM

A new intrinsic (FCARD) has been developed to drive the 7260A as a terminal. Full control of the OMR is provided to the user through a call to the intrinsic. Applications programs may be written in FORTRAN, BASIC, COBOL or SPL. The 7260A may be used for remote applications via full duplex modems or hardwired to the HP 3000 II. FCARD provides the capability of reading marked cards in either ASCII or IMAGE mode.

In the ASCII mode, the card form marks (or punched holes) are interpreted as standard 128 character Hollerith code and the data is transmitted in 7-level ASCII code with even parity. In Image mode, the Optical reader transmits a two character representation of each column of data. This permits all 4096 possible combinations of marks to be read. Switching between ASCII or Image mode can be done under program control.

Documentation of FCARD is included in the 3000 II Intrinsic Manual and in the 7260/3000 Operating Manual (HP P/N 07260-90013) supplied with the 7260A option 300.

Software available for the 7260A is included in the 3000 II Operating System release 05 (date code 1646). In addition to FCARD, software available includes the HP 7260A Application Package. This application package allows a user to simply create a disc file with the contents of his cards and to run an ON LINE diagnostic to verifier operation of the Optical reader.

Further information on the HP 7260A Optical Mark Reader and its operation on 2000 and 3000 systems may be obtained from your local HP sales office. Or feel free to call me at (208) 376-6000.

HP EDITOR/2000 POCKET GUIDE

Dan Jorgenson
HP General Systems

A handy pocket guide for the HP 2000 System editor, *EDITOR/2000*, is available. It contains quick reference information on editor commands, default parameters, line and character position parameters, files, and output format control options.

You may order this pocket guide from your Local HP Sales Office, or use the direct mail order form in the back of the **Communicator**.

Part No.: 22701-90002   **Price: $1.00**

EDITOR'S NOTE

This issue of the **Communicator** does not contain a Software Update section for the HP 2000. The next issue will pick up information on new releases unavailable at the time this issue was published.

documentation

The following tables list currently available customer manuals for HP 2000 Systems products. This list supersedes the list in the last issue of the **Communicator.**

The most recent changes to the tables are indicated for easy reference. Prices are subject to change without notice.

Copies of manuals and updates can be obtained from your local Sales and Service office. The address and telephone number of the office nearest to you are listed in the back of all customer manuals.

Update packages are free of charge. If you require an update package complete the Update Order Form in the back of the **Communicator** and mail the form to:

> Software/Publications Distribution
> 5303 Stevens Creek Blvd.
> Santa Clara, CA. 95050

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manual(s) you need on the Corporate Parts Center form supplied at the back of the **Communicator.**

A few words about documentation terms:

New — A new manual refers only to the first printing of a manual. When first printed, a manual is assigned a part number.

Revised — A revised manual is a printing of an existing manual which incorporates new and/or changed information in its contents. For example, a manual is revised when an update package is incorporated into the manual: the manual gets a new print date and the update package disappears. Note that a revision to a manual effectively obsoletes the previous version of the manual.

Update — An update package is a supplement to an existing manual which contains new and/or changed information. Updates are issued when information must get to customers, yet it is inappropriate to issue a revised manual. An update has no part number; it is automatically included when you order the manual with which it is associated.

PART NUMBER	2000 E	2000 F	HP 2000	MANUAL TITLE	PRICE[†]	PUBLICATION DATE	CURRENT UPDATE
02000-90055		X		2000C/2000F IDF Author's Manual	$ 8.50	1/73	8/74
02000-90080		X		2000E to 2000F Conversion Guide	1.00	4/76	
19665-90001			X	2000/F to 2000 Computer System Upgrade Kit and Conversion Program Manual	2.00	8/76	
19665-90002			X	2000/F to 2000 Computer System Educational Application Upgrades	1.00	2/76	
22687-90001			X	2000 BASIC Reference Manual	10.00	5/76	9/76
22687-90005			X	2000 Operator's Manual	10.00	5/76	9/76
22687-90007			X	2000 System Operator's Pocket Guide	1.50	5/76	
22687-90003			X	BASIC Pocket Guide	1.50	10/76 *R	
02000-90048	X			BASIC/2000 Level E Reference Manual, Timeshared	10.00	9/75	
02000-90049	X			BASIC/2000 Level E System Operator's Manual, Timeshared	5.00	9/74	8/75
5952-4490	X			BASIC/2000 Level E Pocket Guide, Timeshared	0.15	10/74	
02000-90073		X		BASIC/2000 Level F Reference Manual, Timeshared	7.50	12/75	
02000-90074		X		BASIC/2000 Level F System Operator's Manual, Timeshared	10.00	8/76	
5952-4491		X		BASIC/2000 Level F Pocket Guide, Timeshared	0.15	8/75	
24387-90001		X		Basic Analysis and Mapping Program Manual	18.00	6/74	5/75
24387-90002		X		Basic Analysis and Mapping Program Pocket Guide	1.00	6/74	
24384-90001		X	X	College Information System — System Overview	5.00	6/74	9/76
24384-90003		X	X	College Information System Reference Manual	19.00	9/75	9/76
24384-90005		X	X	College Information System — Technical Manual	95.00	5/75	
24383-90001		X		Course Writing Facility Reference Manual	15.00	5/74	
22692-90001			X	Course Writing Facility Reference Manual	16.50	12/75	

PART NUMBER	2000 E	2000 F	HP 2000	MANUAL TITLE	PRICE[†]	PUBLICATION DATE	CURRENT UPDATE
5951-1381		X		DOS-M/2000C Timeshared BASIC File Handler	1.00	5/71	
22701-90001			X	EDITOR/2000 Reference Manual	7.00	9/76	
22701-90002			X	EDITOR/2000 Pocket Guide	1.00	11/76*N	
20352-90001		X		Educational Budget and Accounting System — System Overview	10.00	6/74	
20352-90002		X		Educational Budget and Accounting System — Reference Manual	10.00	3/75	4/76
20352-90003		X		Educational Budget and Accounting System — Technical Manual	75.00	3/75	
20353-90001		X		Educational Payroll System — System Overview	3.50	10/74	
22700-90001			X	FCOPY/2000 Reference Manual	4.50	1/76	
22693-90003			X	HP MATH Curriculum Guide	17.50	7/75	
22693-90002			X	HP MATH Proctor's Manual	6.50	7/75	
22693-90001			X	HP MATH Teacher's Handbook	5.50	7/75	
20310-90007		X		HP MATH Curriculum Guide	20.00	7/74	
20310-90005		X		HP MATH Proctor's Manual	5.00	9/74	
20310-90001		X		HP MATH Teacher's Handbook	5.00	9/74	
22691-90003			X	Instructional Dialogue Facility Author's Manual	13.00	9/75	
22691-90004			X	Instructional Dialogue Facility Author's Pocket Guide	3.00	9/75	
22691-90002			X	Instructional Dialogue Facility Course Developer's	5.00	9/75	
22691-90001			X	Instructional Dialogue Facility Proctor's Manual	6.00	9/75	
20309-90005		X		Instructional Dialogue Facility Author's Pocket Guide	3.50	10/74	
20309-90003		X		Instructional Dialogue Facility Course Developer's Manual	6.00	8/74	
20309-90001		X		Instructional Dialogue Facility Proctor's Manual	10.00	9/74	
22690-90001			X	Instructional Management Facility Proctor's Manual	6.50	9/75	
22690-90002			X	Instructional Management Facility System Manager's Reference Manual	4.50	9/75	
20308-90001			X	Instructional Management Facility Proctor's Manual	7.00	9/74	
20308-90003		X		Instructional Management Facility System Manager's Manual	5.00	10/74	
22687-90009			X	Learning Timeshare BASIC	3.50	5/76	
20243-90001			X	Source Data Entry/2000 Reference Manual	5.00	2/76	
20240-90001			X	Telecommunications Supervisory Package/2000 Manager's Manual	5.00	1/76	
20240-90002			X	Telecommunications Supervisory Package/2000 User's Manual	3.50	1/76	
20311-90001		X		Timeshared Graphics for Tektronix Terminals	7.00	8/74	
20311-90003		X		Timeshared Graphics Plotting Package	5.00	8/74	

*N = New Manual (Refer to the Bulletin section)

*R = Revised Manual

† Prices listed are subject to change without notice.

about the HP 2000

about the HP 2000

training schedule

The schedule for customer training courses on General Systems Division Products is outlined below and in the HP 3000 selection of this publication. Included here are 2000 courses for the 3 month period, January, 1977 through March, 1977.

GENERAL SYSTEMS DIVISION COURSE SCHEDULE

Course Dates and Training Center Location

COURSE NUMBER	COURSE TITLE	LENGTH	PRICE	WESTERN TECHNICAL CENTER – SANTA CLARA	EASTERN TECHNICAL CENTER – ROCKVILLE
22973A	2000 User	5 days	$500	Jan. 24, 1977 Mar. 7, 1977	Scheduled upon sufficient demand.

Registration

Requests for enrollment in any of the above courses should be made through your local HP Sales Office. Your Sales Representative will supply the Training Registrar at the appropriate location with the course number, dates, and requested motel reservations. Enrollments are acknowledged by a written confirmation indicating the training course, time of class, location and accommodations reserved.

Accommodations

Students provide their own transportation, meals, and lodging. The Training Registrar will be pleased to assist in securing motel reservations at the time your Sales Office requests a registration.

Cancellations

In the event you are unable to attend a class for which you are registered, please notify your HP Sales Office immediately. To avoid paying for a reservation which you do not use, we must receive notification of your cancellation no later than 10 working days before the class begins.

EASTERN TECHNICAL CENTER	WESTERN TECHNICAL CENTER
Hewlett-Packard 4 Choke Cherry Road Rockville, Maryland 20850	Hewlett-Packard 5303 Stevens Creek Blvd. Santa Clara, Calif. 95050

A Programming Language
or
Do All Those Funny
Symbols Actually Mean
# Something?

Section I

Whence Cometh APL?

The most recently introduced programming language for the HP 3000 Series II was "A Programming Language" or APL. It was introduced by representatives of Hewlett-Packard Company at the International APL Conference in Ottawa, Canada on September 22, 1976.

APL is a unique computer programming language adaptable to scientific and mathematical applications to meet the informational and analytical needs of a wide range of users. Its growing popularity can be attributed to several factors; specifically APL can help the business/commercial user *as easily as* it can help the scientific user. A professor for a U.S. University has stated "Anybody who actually uses APL becomes a fanatic for it. The detractors generally are people who have not used it themselves."

A prime motivation behind the development of APL was that a person should have access to a wide variety of methods for processing data. The result was a language with about 200 built-in "operators", in the sense that $+,-,*$, and $/$ are operators meaning "add", "subtract", "multiply" and "divide" in other languages. In addition to these, APL also gives access to logarithms in any base ($\circledast$), minimums and maximums ($\lfloor$, $\lceil$), random numbers (?), factorials (!), built in ascending and descending sorts ($\blacktriangle$, $\blacktriangledown$) and a vast assortment of other operations, including many powerful ways of manipulating tables or matrices. Probably one of the most significant advantages of APL over other languages is the fact that all of these operations extend with ease to work with vectors (lists of numbers), matrices (tables) and higher order arrays.

From a user's point of view, though, the advantages of APL over other conventional programming languages are very real. The language can be taught and used in small pieces.

After five minutes of instruction, a student can be trying simple operations at the keyboard — and what he doesn't know won't hurt him. Any erroneous operation he might try will result in an error message, no more.
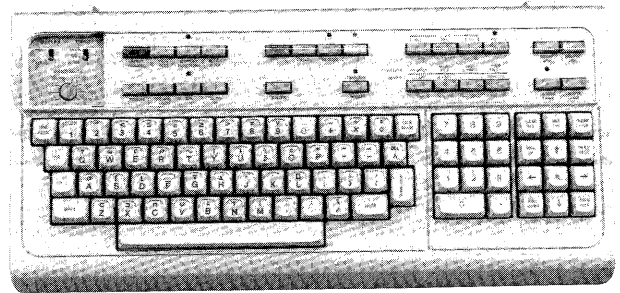
The HP 2641A APL keyboard. Special symbols, while complex looking, are easily learned.

Program development using APL can actually be fun. The language has a built-in "CALCULATOR" mode, which allows a user to type in any valid APL expression and get immediate results. It is very easy to try out alternative methods of performing operational tasks and the vast array of operators make it incredibly simple to do complex operations, and since data declarations are not required APL is ideal for an on-line, interactive approach to applications program development.

Some users see a very real decrease in the time necessary to design and implement a system using APL as opposed to languages such as FORTRAN or COBOL — as much as 5 to 10 times. This is usually attributable to the code compactness of APL. What might take 10 lines in FORTRAN could be done in 1 line in APL.

The examples shown below can illustrate this fact, using a very simple program which finds the sum of a string of numbers that you input.

Basic	Fortran	APL
10 DIM A(100) 20 READ N 30 LET S=0 40 FOR I=1 TO N 50 READ A(I) 60 LET S=S+A(I) 70 NEXT I 80 PRINT S 90 DATA ———— ———— XXX END	DIMENSION A(100) READ (5,10)N 10 FORMAT (13) READ (5,20)(A(I),I=1,N) 20 FORMAT (8E10,3) S=0.0 DO 30 I=1,N · 30 S=S+A(I) WRITE (6,40)S 40 FORMAT (E12,3) END	+/□

Figure 1. Language examples

Users of BASIC or FORTRAN will readily understand the intent of those two programs. They are set up to first read a number which tells how many numbers there are going to be in the list, after which they actually add up the list.

The APL version looks a little different. Since APL actually operates from right to left, the □ is looked at first. This merely is an input request to the user. He would type in a string of numbers and then hit a carriage return. APL then sees +/, which actually means "add up all the numbers you just got and print out the result". Simple enough?

Yet, with all of this power, where can APL find a niche in the business world? Since the whole thrust of the language is mathematical, the immediate (and traditional) association with scientific work is understandable. But with the proper approach, the properties of APL can be profitably brought to bear upon business problems.

The applications must be chosen with care. Even with the availability of an external file handling capability in APL, it is not well suited to problems dealing with massive amounts of input or output, or where on-line interactiveness is not needed. APL is much more at home in a setting where I/O is modest, and where user interaction and feedback are among the most important requirements. Such applications might include performance analysis, financial planning, business simulation, as well as certain types of management information systems.

Many applications deal with tables of numbers (or matrices). APL can handle these with ease. The simple example below will demonstrate this.

A sales manager of a small company has four product lines to sell and five sales people. The sales revenues and the expenses of these people are as follows:

REVENUES

	JONES	SMITH	WALL	HARRI	HALL
SHOES	190.00	140.00	1926.00	14.00	143.00
HATS	325.00	19.00	293.00	1491.00	162.00
PANTS	682.00	14.00	852.00	56.00	659.00
BOOTS	829.00	140.00	609.00	120.00	87.00

EXPENSES

	JONES	SMITH	WALL	HARRI	HALL
SHOES	120.00	65.00	890.00	54.00	430.00
HATS	300.00	10.00	23.00	802.00	235.00
PANTS	50.00	299.00	1290.00	12.00	145.00
BOOTS	67.00	254.00	89.00	129.00	76.00

This data was entered into an APL workspace as follows:

REVENUES←4 5ρ190 140 1926 14 143 325 19
    293 1491 162,□
□:
    682 14 852 56 659 829 140 609 120 87

The "←" is an assignment just like "=" in BASIC. 4 5 ρ means that the data following is to be shaped into a table with four rows and five columns. The □ at the end of the first line tells APL there is more data to come. The □: was typed by APL as a request for the additional data.

By typing the variable name, REVENUES, APL will display it.

REVENUES

190	140	1926	14	143
325	19	293	1491	162
682	14	852	56	659
829	140	609	120	87

The data for Expenses was entered in the same way.

EXPENSES

120	65	890	54	430
300	10	23	802	235
50	299	1290	12	145
67	254	89	129	76

The sales people are paid a commission of 6.2% on the difference between the revenues generated from a product line and the expenses. However, if a salesperson lost money his commission was zero, i.e. the company wasn't paid

back. An APL expression to calculate the commission and save it in a variable called COMMISSION is as follows:

COMMISSION←.062x0⌈REVENUES-EXPENSES

Since APL is executed from right to left we start at the right hand side of the expression to interpret it. First the Expenses table is subtracted from the REVENUES Table. Then all the negative numbers are replaced with zeros. This is done using the maximum operator, ⌈ (A⌈B means take the maximum of the two). The result is multiplied by .062, the commission rate and assigned to the variable COMMISSION.

COMMISSION

4.34	4.65	64.232	0	0
1.55	.558	16.74	42.718	0
39.184	0	0	2.728	31.868
47.244	0	32.24	0	.682

PROFITS can be calculated by adding COMMISSION to EXPENSES and subtracting the results from REVENUES.

PROFITS←REVENUES-EXPENSES+COMMISSION

PROFITS

65.66	70.35	971.768	−40	−287
23.45	8.442	253.26	646.282	−73
592.816	−285	−438	41.272	482.132
714.756	−114	487.76	−9	10.318

To find the PROFITS by product line the APL operator +/ plus reduction is used. This will sum across the columns of a table.

+/PROFITS

780.778	858.434	393.22	1089.834

+⌿ gives the sums across the rows or the profits by sales person.

+⌿PROFITS

1396.682	−320.208	1274.788	638.554	132.45

Since APL executes from right to left either of these operators can be used to calculate total profit

+/+⌿PROFITS

3122.266

It is left to the reader to write APL expressions to do the following:

a. Calculate total commission per sales person.

b. Calculate total revenues per product line.

c. Calculate total revenues per sales person.

d. Calculate total revenues.

e. Calculate total expenses.

f. Round commission to two decimal places.

If you are worried about the difficulty of learning the language, rest assured that all of the 200 operators can be learned and it is rare that one needs to know all of them anyway. The previous example only used seven:

ρ	—	**reshape**
□	—	**request input**
−	—	**subtract**
⌈	—	**maximum**
X	—	**multiple**
+/ ⎫ +⌿ ⎭		**plus reduction**
←	—	**assign**

As a matter of fact, many colleges and universities, as well as some public secondary schools, are teaching APL as a "first" programming language to students. It is an ideal language for such purposes, because you just do not get caught up in lots of syntax formats, word misspellings and the associated miscellaneous problems that occur when learning your first programming language.

The developer of APL, Kenneth Iverson, expressed it well when he said:

"The power of an adequate programming language amply repays the effort required for its mastery"

And though it might take a number of months to "master" APL, it certainly does not take a week of instruction before someone can use it.

So, if you have a need for a very interactive approach for learning or development, see how APL might be able to solve your problems.

SECTION II. APL/3000.

For those readers familiar with APL, the first section of this article merely stated things you already know about the language. It might be interesting for you to hear more about APL/3000, the newly released version on the HP 3000 Series II Computer System.

APL/3000 is basically an extension of APLSV, IBM'S standard version on the 370 line of computers. But we have made many extensions and improvements to the APL language in general:

● VIRTUAL WORKSPACES — As you know, workspace size in the past has severely limited the use of APL in some applications. Even the largest mainframes allowed a fixed workspace. With APL/3000, the size is limited only by the amount of on-line disk storage available.

● DYNAMIC COMPILER — APL has always been implemented as an interpreter. APL/3000, on the other hand, actually compiles and saves the code necessary to execute an APL function. When the function is called again, the compiled code is used (after first checking to see if the code is still usable on the current data types and ranks) which means that there can be a significant speed-up of execution on subsequent runs of a function. Compilation allows a certain amount of code optimizing. For example, if A, B and C were each 1000 element vectors,

$$5 \uparrow A \times B \div C$$

it would cause an interpreter to do 1000 divides, 1000 multiplies, and then give you the first 5 elements. APL/3000 on the other hand, would only do 5 divides, 5 multiplies, and be finished!

● APLGOL — A structured programming extension to APL, which uses ALGOL-like control structures, with the power of APL operators.

● FILE HANDLING — APL/3000 has full access to the MPE file intrinsics, which means that from within APL, you can use disc files, tape files, card readers and other devices. As a part of this facility, it is also possible to issue MPE commands such as file equations which increase the flexibility of the entire APL subsystem.

● EXTENDED CONTROL FUNCTIONS — A set of functions which allows the user to organize the relationship between user defined functions in a flexible manner and obtain access to variables local to different functions.

And in addition to these five main extensions APL/3000 has added numerous systems functions and variables.

This is the first time that an APL of such power is available on a machine the size of the HP 3000 and we will be glad to provide more information about APL/3000 and its application to meet our users specific requirements.

The Computer Science Department at Yale University, under the direction of Professor Alan Perlis, is the test site for the first small, economical computer system with a complete APL software. According to Perlis, the Hewlett-Packard APL 3000 is "fully equivalent" to the APL on large mainframes.

The intricate APL characters of a typical business program appear crisp, clean and clear on the high resolution screen of the 2641A.

software tips

about the HP 3000

SEGMENTATION IN COBOL

COMPILE TIME SEGMENTATION

Greg Gloss
HP General Systems

To effectively use the Segmenter with COBOL programs, you should first understand how the COBOL compiler generates code segments. A COBOL main program produces two or more code segments depending on the number of sections in the Procedure Division. The first segment contains one unit which is the outer block for your program. The unit name is formed by appending an apostrophe to the name specified in the PROGRAM-ID paragraph. The outer block initializes the run-time data area for the main program.

By using sections with different priority numbers in your main program, you can segment your program into several code segments. If you do not segment your program, it will be put into one segment. The program shown below produces three segments, one for initialization and two for the Procedure Division code.

```
001000   IDENTIFICATION DIVISION.
001100   PROGRAM-ID. MAIN.
001200   ENVIRONMENT DIVISION.
001300   DATA DIVISION.
001400   PROCEDURE DIVISION.
001450   FIRST-SEC SECTION.
001500   START.
001600       DISPLAY "START OF MAIN PROGRAM".
001700       CALL "SUB".
001720   SECOND-SEC SECTION 02.        Priority
001730   START-2.                       Number
001750       CALL "SUBA".
001800       DISPLAY "END OF MAIN PROGRAM".
001900       STOP RUN.
```

DATA AREA IS %000271 WORDS.
CPU TIME = 0:00:01. WALL TIME = 0:00:05.
END COBOL/3000 COMPILATION. NO ERRORS. NO WARNINGS.

Each COBOL subprogram produces two segments, one for initialization and one for the actual code. The initialization procedure name and its corresponding segment name are formed by appending an apostrophe to the name specified in the PROGRAM-ID paragraph. Note that subprograms cannot be segmented. The entire subprogram code must go into one segment.

USING THE SEGMENTER

In the past, there have been several restrictions for using the Segmenter to move COBOL subprograms around. Beginning with versions B.02.04 and C.01.02 of the COBOL compiler, only the following restrictions still apply:

1. The outer block must be the first program unit presented at PREP time which allocates Secondary DB storage. The base address for the main program data area must be DB+0.

2. The outer block and non-dynamic subprograms cannot be put into an SL.

When studying the following examples, remember that the Segmenter inserts new procedures and new segments at the top of the list. Therefore, you may have to take action to put things in the proper order.

The following items allocate Secondary DB storage and therefore cannot precede the outer block:

1. Non-dynamic COBOl subprograms.

2. SPL procedures with TRACE, EXTERNAL, or OWN variables.

3. FORTRAN procedures with DATA, COMMON, LABELED COMMON, or TRACE variables.

Dynamic COBOL subprograms and other procedures which do not use global storage may precede the outer block.

Consider the USL file, shown below, which was produced by compiling a main program containing two Sections in the Procedure Division.

```
USL FILE DOCUSL.COBOL.LANG
MAIN                                Outer Block Designation
    MAIN'            40   OB A C N      Main Program
SECONDSEC02'                           Initialization
    SECONDSEC02'    31  P  A C N R
FIRSTSEC00'
    FIRSTSEC00'     33  P  A C N R

FILE SIZE       377600
DIR. USED          166       INFO USED         377
DIR. GARB.           0       INFO GARB.          0
DIR. AVAIL.      37212       INFO AVAIL.    337401
```

Now, suppose you have compiled a non-dynamic subprogram into another USL file and want to move it into the

master USL file. The first step is to COPY the two subprogram segments into the master USL file using the Segmenter.

```
USL DOCUSL
AUXUSL DOCAUX
COPY SUB          (Copy subprogram initialization)
COPY SUB'         (Copy dynamic subprogram body)
LISTUSL
```

USL FILE DOCUSL.COBOL.LANG

SUB'						
SUB'	145	P	A	C	N	R
SUB						
SUB	466	P	A	C	N	R
MAIN						
MAIN'	40	OB	A	C	N	
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R
FIRSTSEC00'						
FIRSTSEC00'	33	P	A	C	N	R

However, this leaves the USL file in an improper condition since a non-dynamic subprogram precedes the Outer Block. There are two ways to correct this situation. The first way is to move the Outer Block to the first segment by using a NEWSFG command as shown below.

```
NEWSEG SUB',MAIN'
PURGERBM SEGMENT,MAIN
LISTUSL
```

USL FILE DOCUSL.COBOL.LANG

SUB'						
MAIN'	40	OB	A	C	N	
SUB'	145	P	A	C	N	R
SUB						
SUB	466	P	A	C	N	R
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R
FIRSTSEC00'						
FIRSTSEC00'	33	P	A	C	N	R

FILE SIZE	377600			
DIR. USED	331	INFO USED	2045	
DIR. GARB.	7	INFO GARB.	0	
DIR. AVAIL.	37047	INFO AVAIL.	335733	

The second way is to use the NEWSEG command to create a new segment for the Outer Block as shown below.

```
NEWSEG NEWMAIN,MAIN'
LISTUSL
```

USL FILE DOCUSL.COBOL.LANG

NEWMAIN						
MAIN'	40	OB	A	C	N	
SUB'						
SUB'	145	P	A	C	N	R
SUB						
SUB	466	P	A	C	N	R
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R
FIRSTSEC00'						
FIRSTSEC00'	33	P	A	C	N	R

The USL file can now be prepared.

If you want to add a dynamic subprogram or a procedure in another language (such as SPL or FORTRAN) which does not use global storage, the COPY command can be used without any further action.

```
COPY SUBA'         (Copy subprogram initialization)
COPY SUBA
LISTUSL             (Copy dynamic subprogram body)
```

USL FILE DOCUSL.COBOL.LANG

SUBA						
SUBA	327	P	A	C	N	R
SUBA'						
SUBA'	211	P	A	C	N	R
NEWMAIN						
MAIN'	40	OB	A	C	N	
SUB'						
SUB'	145	P	A	C	N	R
SUB						
SUB	466	P	A	C	N	R
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R
FIRSTSEC00'						
FIRSTSEC00'	33	P	A	C	N	R

Now, suppose you want to combine some segments together. You can use the NEWSEG command to combine initialization procedures, main program modules and subprogram compilations.

```
NEWSEG SUB,SUB'
PURGERBM SEGMENT,SUB'
NEWSEG SUBA',SUBA
PURGERBM SEGMENT,SUBA
LISTUSL
```

USL FILE DOCUSL.COBOL.LANG

SUBA'						
SUBA	327	P	A	C	N	R
SUBA'	211	P	A	C	N	R
NEWMAIN						
MAIN'	40	OB	A	C	N	
SUB						
SUB'	145	P	A	C	N	R
SUB	466	P	A	C	N	R
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R
FIRSTSEC00'						
FIRSTSEC00'	33	P	A	C	N	R

You can also move a main program segment into another segment.

```
NEWSEG SUBA',FIRSTSEC00'
PURGERBM SEGMENT,FIRSTSEC00'
LISTUSL
```

USL FILE DOCUSL.COBOL.LANG

SUBA'						
FIRSTSEC00'	33	P	A	C	N	R
SUBA	327	P	A	C	N	R
SUBA'	211	P	A	C	N	R
NEWMAIN						
MAIN'	40	OB	A	C	N	
SUB						
SUB'	145	P	A	C	N	R
SUB	466	P	A	C	N	R
SECONDSEC02'						
SECONDSEC02'	31	P	A	C	N	R

FILE SIZE	377600			
DIR. USED	461	INFO USED	3154	
DIR. GARB.	37	INFO GARB.	0	
DIR. AVAIL.	36717	INFO AVAIL.	334624	

Since COBOL code modules no longer have to begin at PB+0, they can now be put into an RL.

616

In summary, the following items can be put into an RL:

    COBOL subprograms
    Procedures in other languages
    Subprogram initialization procedures
    Main Program modules

The following cannot be put into an RL:

    Outer Block (Main program initialization)

The following can be put into an SL:

    Dynamic subprograms
    Subprogram initialization routines
    Procedures in other languages without global data

The following cannot be put into an SL:

    Outer Block (Main program initialization)
    Non-dynamic subprograms
    Procedures in other languages with global data

FORTRAN: INPUT/OUTPUT OF COMPLEX NUMBERS

Madeline Lombaerde
HP General Systems

It is possible to input complex numbers in a variety of ways. Using formatted input, the F or G descriptor may be used:

```
COMPLEX A, B
READ (5,500) A, B
500 FORTRAN (2F10.2, 2G12.4)
```

Normal rules for number fields should be followed.

With free field input, the use of parentheses around the complex number pair is optional.

```
COMPLEX A, B
READ (5, *) A, B
```

will accept 3.7, 4.2, (6.1, 8.2) as input

where
    A = (3.7, 4.2)

and
    B = (6.1, 8.2)

Similarly, output may be free field (DISPLAY or WRITE (6, *)) or formatted. An example of formatted output would be

```
COMPLEX A, B
.
.
.
WRITE (6,600) A, B
FORMAT ("A = ", 2F10.2, "B = ", 2G12.4)
```

Free field output follows the same rules for output of floating point numbers (2 will be output for each complex number.)

CALLING SPL FROM RPG AND CALLING COBOL FROM SPL

Bruce Campbell
HP Neely Sales Region

I.   Calling SPL from RPG using RLABL and EXIT. The RPG compiler generates the equivalent of a global byte pointer declaration for fields used in an RLABL operation. For example,

    RLABL   XYZ

generates the equivalent of

    global byte pointer XYZ;

To access parameters declared with the RLABL operation from an SPL procedure, the parameter should be declared in the local declaration section of the procedure as "EXTERNAL BYTE POINTER". For example, given the following RPG code,

    RLABL   XYZ
    EXIT EXTPRO

The SPL procedure would start:

    PROCEDURE EXTPRO;
    BEGIN
        EXTERNAL BYTE POINTER XYZ;

Notice the following points:

1.   It does not matter whether the RPG field named in an RLABL operation is alpha or numeric (i.e. packed). Both are referenced as external byte pointers from a called SPL procedure.

about the HP 3000

about the HP 3000

2. Parameters passed to SPL procedures by RLABL are not defined in the procedure head, but only in the local declarations.

Attached is an RPG program "RPGDATE" that calls an SPL procedure "MDYCON" and passes it two fields "MDYRZZ" and "JDATZZ" thru the RLABL operation.

II. Calling COBOL from SPL.
There are three things to keep in mind:

1. Compile the COBOL program with the "DYNAMIC" option in the $CONTROL statement. This forces the COBOL compiler to allocate all data division storage Q-RELATIVE.

2. Specify the parameters passed to the COBOL program in both the linkage section and in the using option of the procedure statement. For example:

   LINKAGE SECTION.
   01 PARAM1 PIC S9(5) COMP-3.
   01 PARAM2 PIC S9(4) COMP.
   PROCEDURE DIVISION USING PARAM1 PARAM2.

The parameters must be specified in the using option of the procedure division statement in the same order that they are given in the SPL call to the program. Also, the parameters must be declared in the linkage section at the 01 LEVEL.

3. All parameters passed to a COBOL program must be word-addressed since the COBOL compiler always generates word address for external data items (as distinguished from byte-addresses). Also, the COBOL compiler always starts 01-LEVEL COMP-3 fields in the left-hand byte of a word. This means that the declaration

   01 FIELD PIC S9(5) COMP-3.

which uses two words. Has a slack byte in the right-most byte of those two words. COBOL always handles parameters by reference, except file number is passed from COBOL by value.

III. Examples.
Attached are three programs.

1. RPGDATE: An RPG program that call an SPL routine called "MDYCON" and makes two packed decimal data fields available to it.

2. MDYCON: An SPL program that calls a COBOL routine called "MDYCONC". MDYCON does not do anything except take care of the addressing incompatibility between the RPG byte pointers and the COBOL word addresses.

3. MDYCONC: Translates the MM/DD/YY date passed from the RPG program to Julian format, and returns the Julian date to the RPG program.

RPGDATE

```
0001  $TITLE "*** TEST DATE CONVERSION:
      MM/DD/YY TO JULIAN *****"
0002  H

0003  FINFILE    IPEAF 72 72   DISC
0004  FOUTFILE  O  F 72 72    DISC

0005  IINFILE     AA 01
0006  I                              1 60DATEIN

0007  C          RLABL          JDATZZ  50
0008  C          RLABL          MDYRZZ  60
0009  C          Z-ADDDATEIN    MDYRZZ
0010  C          EXIT MDYCON

0011  OOUTFILE  D   01
0012  O                         JDATZZX 10
```

SYMBOL	SZ/TP	ADDR	SYMBOL	SZ/TP	ADDR	SYMBOL	SZ/TP	ADDR
DATEIN	6.0	00374(L)	JDATZZ	5.0	00376(L)	MDYCON	SUBR	

```
NO. SERIOUS ERRORS 000      NO. WARNINGS 000
PROCESSOR TIME=0:00:03;     ELAPSED TIME=0:00:35
```

MDYCON

```
00001000 00000 0  $CONTROL SUBPROGRAM
00002000 00000 0   BEGIN
00003000 00000 1   PROCEDURE MDYCON:
00004000 00000 1    BEGIN
00005000 00000 2    <<
00006000 00000 2    ***ALIGN PARAMETERS ON WORD BOUNDARIES
00007000 00000 2    *** AND PASS WORD ADDRESSES TO COBOL
00008000 00000 2    >>
00009000 00000 2    EXTERNAL BYTE POINTER MDYRZZ,JDATZZ;
00010000 00000 2    LOGICAL ARRAY MDYR(0:1),JUL(0:1);
00011000 00000 2    BYTE ARRAY BMDYR(*)=MDYR,BJUL(*)=JUL;
00012000 00000 2    PROCEDURE MDYCONC(DATE'IN,DATE'OUT);
00013000 00000 2     INTEGER ARRAY DATE'IN,DATE'OUT;
00014000 00000 2     OPTION EXTERNAL;
00015000 00000 2    MOVE BMDYR:=MDYRZZ,(4);
00016000 00021 2    MDYCONC(MDYR,JUL);
00017000 00024 2    MOVE JDATZZ:=BJUL,(3);
00018000 00030 2    END;
00019000 00000 1   END;
```

```
PRIMARY DB STORAGE=%000;      SECONDARY DB STORAGE=%00000
NO. ERRORS=000;               NO. WARNINGS=000
PROCESSOR TIME=0:00:02;       ELAPSED TIME=0:00:29
```

MDYCONC 1

```
001000STITLE "*********CONVERT STANDARD TO JULIAN***********"
001100SCONTROL DYNAMIC,MAP
001200 IDENTIFICATION DIVISION.
001300 PROGRAM-ID. MDYCONC.
001400 ENVIRONMENT DIVISION.
001500 DATA DIVISION.
001600 FILE SECTION.
001700 WORKING-STORAGE SECTION.
001750 01  UNPAK-DATE PIC 9(6).
001800 01  DATE-SUBS.
001900     02 MM PIC 99.
002000     02 DD PIC 99.
002100     02 YY PIC 99.
002200 01  MONTH-DISPLACEMENT-TABLE.
```

618

```
002300     02 MONTH-DISP PIC 999 OCCURS 12 TIMES.
002400 01  QUOT PIC S999.
002500 01  REM PIC S9.
002600 LINKAGE SECTION.
002700 01  MDY-DATE PIC S9(6) COMP-3.
002800 01  JULIAN-DATE PIC S9(5) COMP-3.
002900*****
003000 PROCEDURE DIVISION USING MDY-DATE JULIAN-DATE.
003100 START-MDYCON SECTION.
003200     DISPLAY "RECEIVED  " MDY-DATE.
003300     MOVE "00003105909012015118121224327330 4334"
003400       TO MONTH-DISPLACEMENT-TABLE.
003500     MOVE MDY-DATE TO UNPAK-DATE.
003600     MOVE UNPAK-DATE TO DATE-SUBS.
003800     DISPLAY "Y,M,D:" YY ";" MM ";" DD ";".
003900     IF YY = 0 OR MM = 0 OR MM > 12 OR DD = 0 OR DD > 31 THEN
004100       MOVE 99999 TO JULIAN-DATE
004150       DISPLAY "FLUSHED WITH " JULIAN-DATE
004200       GOBACK.
004300     COMPUTE JULIAN-DATE = YY * 1000.
004400     COMPUTE JULIAN-DATE = JULIAN-DATE + MONTH-DISP(MM).
004500     COMPUTE JULIAN-DATE = JULIAN-DATE + DD.
004600     DISPLAY "SENT BACK " JULIAN-DATE.
004700     DIVIDE 4 INTO YY GIVING QUOT REMAINDER REM.
004800     IF REM NOT = ZERO THEN GOBACK.
004900     IF MM > 2 COMPUTE JULIAN-DATE = JULIAN-DATE + 1.
005000     GOBACK.
```

MDYCONC2

SYMBOL TABLE MAP

LVL SOURCE NAME	BASE	DISPL	SIZE	USAGE	CATEGORY	R	O	D	J
WORKING-STORAGE SECTION									
01 UNPAK-DATE	Q	000174	000006	DISP	DISP-N				
01 DATE-SUBS	Q	000202	000006		GROUP				
02 MM	Q	000202	000002	DISP	DISP-N				
02 DD	Q	000204	000002	DISP	DISP-N				
02 YY	Q	000206	000002	DISP	DISP-N				
01 MONTH-DISPLACEMENT-TABLE	Q	000210	000044		GROUP				
02 MONTH-DISP	Q	000210	000044	DISP	DISP-N		O		
01 QUOT	Q	000254	000003	DISP	DISP-NS				
01 REM	Q	000260	000001	DISP	DISP-NS				
LINKAGE SECTION SECTION									
01 MDY-DATE	LINK	000000	000004	COMP-3	N				
01 JULIAN-DATE	LINK	000000	000003	COMP-3	N				

PMAP

RPG'01	0			
NAME	STT	CODE	ENTRY	SEG
RPG'08	1	0	0	
R'OUT	2			1
R'INT	3			1
R'CALT	4			1
R'CALD	5			1
R'CNTL	6			?
SEGMENT LENGTH		34		
RPG'00	1			
NAME	STT	CODE	ENTRY	SEG
R'OUT	1	0	0	
R'WRITE	6			?
R'CALT	2	33	33	
R'CALD	3	37	37	
MDYCON	7			2
R'INT	4	65	65	
R'ERROR	10			?
I'0001	5	65	110	
SEGMENT LENGTH		150		
RL SEGMENT	2			
MDYCON	1	0	0	
MDYCONC	2	31	31	
C'LIT'FIG'MOVE	4			?
C'DECABS	5			?
MPYD	6			?
C'DISPLAY'INIT	7			?
C'DISPLAY'L	10			?
C'DISPLAY'ID	11			?
C'DISPLAY'FIN '	12			?
DIVD	13			?
MDYCONC'	3	1154	1154	
SEGMENT LENGTH		1420		

PRIMARY DB	131	INITIAL STACK	2000	CAPABILITY	600
SECONDARY DB	265	INITIAL DL	0	TOTAL CODE	1624
TOTAL DB	416	MAXIMUM DATA	?	TOTAL RECORDS	17
ELAPSED TIME	00:00:05.042			PROCESSOR TIME	00:01.740

bulletins

PRE-SERIES II HP 3000 SUPPORT

Dave Sanders
HP General Systems

In order to communicate HP's position and plans relative to pre-Series II HP 3000's, the following letter has been recently sent to all owners of pre-Series II HP 3000's. Because our mailing list may not be 100% complete or accurate, the letter is being reproduced here as well.

HEWLETT **hp** PACKARD

GENERAL SYSTEMS · 5303 Stevens Creek Blvd., Santa Clara, California 95050. Telephone 408-249-7020, TWX 910-338-0215

November 30, 1976

Attention: HP 3000 Data Processing Manager

Dear Sir:

Now that the HP 3000 Series II has been announced, I thought it would be appropriate to communicate with users of pre-Series II HP 3000 systems in order to clarify Hewlett-Packard's position and plans relative to these computers.
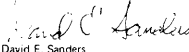
Although the pre-Series II HP 3000 (or 3000 CX as it was sometimes called) is no longer in new production, we at HP definitely view it as an on-going activity. We will continue to offer on-site maintenance support for both the hardware and software, as we have done in the past. The General Systems Division will continue issuing new releases of MPE-C which improve reliability or correct problems of which we become aware. We will also continue to correct problems with software subsystems which run on pre-Series II systems (EDIT/3000, SORT/3000, RPG/3000, etc.). I would like to emphasize that we at Hewlett-Packard definitely do not view the pre-Series II HP 3000 as an "obsolete" product.

In order to insure that the support of our pre-Series II customers gets the appropriate level of management attention, HP has decided to establish a new organization within the General Systems Division, reporting directly to the General Manager, to be responsible for continuing to improve the level of satisfaction of these customers. Ed McCracken, the General Manager of the General Systems Division, has asked me to head up this new organization which will be specifically responsible for software and system development and support of the pre-Series II systems, as well as for the marketing of the pre-Series II-to-Series II upgrade. This organization will therefore be responsible not only for providing our pre-Series II customers the on-going support needed to continue to operate their machines, but also for providing them with the information they may need if their long-term plans call for upgrading to a Series II.

This team of hardware and software engineers is already in place, and has accepted full development and support responsibility for MPE-C. This group has the charter to implement both enhancements and reliability improvements. Most of our resources today are concentrated on maintaining and improving reliability of MPE-C. In the future we also hope to be able to offer enhancements based on your needs.

Our overall objective with our new organization is to improve the level of satisfaction of our pre-Series II customers. We hope to do this by focusing our efforts on continuing to improve the quality of support you receive for your current system. Our software team will be making a special effort to respond to your problems in a professional and timely fashion. We hope you will be pleased with the results.

Yours truly,

David E. Sanders
Manager, Pre-Series II Operations
General Systems Division

DES:lc

COBOL/3000 OBJECT CODE IMPROVEMENTS

Greg Gloss
HP General Systems

Included in versions B.02.03 and C.01.01 of the COBOL/3000 Compiler were some improvements to the object code generated for certain constructs. If you have programs which run a long time you will probably want to recompile them using one of these versions. Although the amount of

improvement will depend on your individual program, we have reduced one program from 5 hours to 90 minutes. The constructs which will be improved are:

1. If *identifier-1 relational-operator identifier-2* where *identifier-1* and *identifier-2* are alphabetic or alphanumeric items of the same length (EXAMPLE: IF A EQUALS B . . .)

2. If *identifier-1 relational-operator literal-2* where *identifier-1* is an alphabetic or alphanumeric item the same length as the literal (EXAMPLE: IF A EQUALS "ABC" . . .)

3. MOVE *figurative-constant* TO *identifier-1* (EXAMPLE: MOVE SPACES TO A)

4. MOVE *literal* TO *identifier-1* (EXAMPLE: MOVE "ABC" TO A)

5. MOVE ALL *literal* TO *identifier-1* (EXAMPLE: MOVE ALL "A" TO A)

6. MOVE *identifier-1* TO *identifier-2* where *identifier-1* and *identifier-2* are COMP items of the same length and characteristics (EXAMPLE: MOVE I TO J)

7. SEARCH and SEARCH ALL statements

In addition to reduced run times, you will also notice some reduction in the size of the code generated by the compiler if you have these constructs in your programs.

HP 3000 COMPUTER SYSTEMS FORTRAN GUIDE

Sandy Martensen
HP General Systems

The new HP Computer Systems FORTRAN Pocket Guide is now available to both 3000 pre Series II and Series II users. The manual contains statement and compiler command summaries, parameter and format information, and tables of functions. It also includes quick reference information about MPE commands used to compile, prepare,

and execute FORTRAN programs and default MPE file parameters. The error conditions which can be specified in a Trap statement are listed.

The manual part number is 32102-90002 and the price is $1.50.

APL/3000 REFERENCE MANUAL AND POCKET GUIDE AVAILABLE

Hal Goodwin
HP General Systems

A reference manual and a pocket guide are available for APL/3000.

The reference manual explains APL/3000 primitive functions, mixed functions, system commands, system variables, and system functions. Examples are provided which illustrate usage. Also covered are the APL/3000 Editor and APLGOL.

Title: APL/3000 Reference Manual
Part #: 32105-90002
Price: $12.00

The pocket guide contains syntax requirements and brief functional descriptions of the same material covered in the manual.

Title: APL/3000 Pocket Guide
Part #: 32105-90003
Price: $1.50

SPL POCKET GUIDE 32100-90001

For SPL users, this new pocket guide provides a handy reference for statements, assemble formats and instruction mnemonics. The pocket guide is 35 pages long and contains many other items of information such as computer and MPE/3000 commands, parameters, and machine instructions.

software updates

about the HP 3000

Each issue of the **Communicator** provides you with information pertinent to the status of 3000 software products including the latest software changes and enhancements.

The 3000 software updates described in this issue relate to the following products:

PRODUCT	3000 PRE SERIES II	3000 SERIES II	NUMBER	UPDATE AND FIX LEVEL	MIT TAPE DATE CODE
MPE C	X		32000C	00.14	1646
MPE II		X	32002A	00.05	1646
2780/3780 Emulator	X		30130B	02.00	1646
2780/3780 Emulator		X	30130C	00.02	1646
BASIC/3000	X	X	32101B	00.05	1646
BASICOMP/3000	X	X	32103B	00.02	1646
COBOL/3000	X		32213B	02.03	1646
COBOL/3000	X	X	32213C	01.01	1646
CROSSLOADER	X	X	32226A	02.00	1636
Fortran/3000	X	X	32102B	00.05	1646
QUERY/3000	X	X	32216A	03.02	1646
RTE Programmable Controller		X	30301B	00.01	1636
SORT/3000	X	X	32214B	01.02	1646

Where changes in documentation are indicated, updates to the appropriate manuals will be printed. This information is provided simply as a temporary measure.

about the HP 3000

MPE 32000C.00.14 AND MPE 32002A.00.05

This article describes MPE 32000C.00.14 and MPE 32002A.00.05 as incorporated into the 3000 pre Series II and 3000 Series II MIT tapes date coded 1636 and 1646. The information in the article is organized as follows.

1. Modules modified for MPE C.00.14 and MPEII A.00.05

2. Supported Utility Module changes for MPEII A.00.05

3. List of problems solved in MPE C.00.14 and MPEII A.00.05

4. Enhancements to MPEII A.00.05

5. Known problems in MPEI C.00.14 and MPEII A.00.05

1A MODULE CHANGES	C.00.XX						MPE C.00.14								
MODULE		1	2	3	4	5	6	7	8	9	10	11	12	13	14
INITIAL	0	X		X	X			X	X	X	X	X	X	X	
SYSDUMP	1	X	X	X			X		X	X			X		X
SEGPROC	2	X	X				X			X	X				
SEGDVR	3												X		
DISPATCH	4			X			X				X	X	X		
LOAD	5		X										X		
MAPP	6					X									
UCOP	7	X													
DEVREC	8														
PROGEN	9	X						X	X	X		X			
IMIN	10						X		X		X	X		X	
EXIN	11	X	X	X		X	X	X		X		X			X
LOG	12	X									X				
IOPTRD0	13														
IOPTPH0	14						X		X						
IOPLOT0	15														
IOMDISK0	16			X			X	X	X						
IOFDISK0	17			X			X	X							
IOTAPE0	18				X			X							
IOLPRT0	19											X	X		
IOCDRD0	20		X				X								
IOCLTTY0	21														
IOTERM0	22											X			
IOCDPN0	23														
IOPRPN0	24					X	X				X	X			
IOMDISK1	27	X		X				X	X	X	X				
PFAIL	30		X	X	X										
FILESYS	50	X	X	X	X	X	X	X	X	X	X	X	X	X	X
COMM' INT	51	X		X			X					X	X		X
STORE/RESTORE	52			X	X			X	X	X			X		
DIRC	53									X				X	
ALLOCATE	54		X		X			X				X	X		
DISKSPC	55	X													
MMCORER	56						X		X			X			
MMDISKR	57											X			X
ABORTRAP	58						X	X		X		X		X	
MESSAGE	59								X		X	X			
CROUTINE	60			X	X						X	X			
IOUTILITY	61	X		X	X			X	X	X					X
TTYINT	62		X	X	X			X		X					
PCREATE	63	X									X				
MORGUE	64			X							X	X			
PROCMAIL	65														
PINT	66						X		X	X		X			
DATASEG	67	X								X					
IOPM	68		X			X						X	X		
CHECKER	69														
UTILITY	70	X	X	X		X				X			X	X	
SEGUTIL	71	X		X				X		X			X		X
LOADER1	72		X	X				X		X			X		X
RINS	73						X					X			
JOBTABLE	74	X													
DEBUG	75	X													
NURSERY	76			X											
SYSDPLY	77						X						X		
FIRMWARESIM	78	X		X	X					X	X	X			
SPOOLING	79			X	X			X	X	X	X	X			
SPOOLCOMS	80	X					X	X		X	X		X		
MESSAGE CAT	–			X			X	X	X	X			X	X	

1B. MODULE CHANGES	A.00.XX	MPEII	A.00.05		
MODULE		1	2	3	4	5
INITIAL	0		X	X	X	
SYSDUMP	1		X	X	X	X
SEGPROC	2		X			
SEGDVR	3		X			
DISPATCH	4		X			
LOAD	5		X			
UCOP	7		X		X	
DEVREC	8				X	
PROGEN	9		X	X	X	X
ININ	10				X	
MEMLOGP	11		X		X	X
LOG	12				X	
IOPTRD0	13		X	X		
IOPTNP0	14		X	X	X	X
IOPLOT0	15					
IOMDISC0	16					
IOFDISC0	17					
IOTAPE0	18					X
IOLPRT0	19		X		X	
IOCDRD0	20					X
IOTERM0	22		X	X	X	X
IOPRPN0	24					
IOREM0	25					
IOMDISC1	27				X	X
PFAIL	30				X	X
FILESYS	50		X	X	X	X
COMM'INT	51			X	X	X
STORT/RESTORE	52					
DIRC	53					
ALLOCATE	54					X
DISCSPC	55					
MMCORE	56				X	
MMDISKR	57		X	X	X	
ABORTTRAP	58		X			X
MESSAGE	59				X	
CROUTINE	60					
CLOCKIO	61					X
NRIO	62		X	X		X
PCREATE	63		X			X
MORGUE	64					
PROCMAIL	65					
PINT	66		X			X
DATASEG	67		X		X	
CRIO	68		X	X	X	X
CHECKER	69					
UTILITY	70				X	X
SEGUTIL	71		X			
LOADER1	72		X			
RINS	73	X		X		
JOBTABLE	74					
DEBUG	75				X	
NURSERY	76				X	
STKDUMP	77					
FIRMWARESIM	78					
SPOOLING	79					X
SPOOLCOMS	80					X
MESSAGE CAT	–					

2. SUPPORTED UTILITY MODULE CHANGES FOR MPEII A.00.04					
MODULE	1	2	3	4	5
DISKEDT2					
DPAN2		X	X	X	X
FREE2					
LISRDIR2					
LISTEQ2					
LISTLOG2		X	X		
PATCH2					
MEMLOGAN					
MEMTIME					
SADUTIL					
SLPATCH					

3A. PROBLEMS SOLVED IN MPE C.00.14

a. When trying to defer a spoofle that was causing a special forms request, it was lost if the following sequence of console commands were used:

```
=SPOOL (LDN),WAIT
=REPLY (PIN),0
```
S.P.R #1179

about the HP 3000

b. The command SETMSG has been corrected to allow for a lower case parameter.
S.P.R. #97

c. An inconsistent response to the purging of a non-existent temporary file is no longer given. In addition such a condition is no longer a fatal error in batch.
S.P.R. #872

d. The error message created in batch, when the output priority provided on a file command is out of bounds, is no longer misleading.
S.P.R. #729

e. The intrinsic CTRANSLATE now returns valid condition codes.
S.P.R. #606

f. The intrinsic EXPANDUSLF opened the new USL file with the same access capability as the old USL file regardless of its capability. EXPANDUSLF now insures that the new USL file has read access capability.
S.P.R. #282

g. The check for overflow in the intrinsic BINARY has been corrected.

h. The FILE command now properly handles invalid file names.

i. SYSDUMP no longer generates CR,LF'S in the interactive dialog when in batch mode.

j. Comments are now allowed on directives to SYSDUMP. Comments must be bracketed in the same manner as specified for SPL.

k. The sequence of events in releasing virtual memory has been modified. The events as previously executed could cause clobbered stacks under a heavy workload.

l. Large number of I/O requests to disc by a single process would lock out other user requests to the same device. This problem has been corrected.

m. The loader was created with the default MAXDATA size. It is now created with MAXDATA=16K.

3B. PROBLEMS SOLVED IN MPEII A.00.05

a. A bug introduced in MEMLOGP of the last MIT has been fixed. The problem was related to a spurious FOPEN failure when attempting to create a new MEMLOG file.

b. CRIO has been modified to keep the system informed that it had completed MAKEPRESENT successfully. Under certain conditions, a S.F. 127 was encountered without this fix.

c. IOCRD0 has been modified to send a master clear to the controller after detecting an I/O error. Without this, the controller could get hung up in a power ON/OFF sequence.

d. ABORTRAP has been modified such that KILLPROG now sets the hard kill bit.

e. IOTERM0 has been corrected as follows:

1. The data read prior to BREAK is now discarded on FCLOSE.

2. CR/LF is not issued on timed out reads.

3. The extra byte in odd byte reads will keep its original content. It will no longer be blank or zero filled.

4. A problem was resolved of potentially losing data when doing an I/O without wait.

f. GETREC of the FILESYS was modified to correct a problem of occasionally losing a byte of information from the system buffer.

g. A timing problem between =REPLY and =SPOOL...that was causing the deletion of SPOOFLES has been corrected.

h. The following problems have been resolved in CRIO.

1. On block mode input, in transparent mode, the system could hang up waiting for a carriage return. This problem has been corrected.

2. The definition of TAPEMODE for 264X is modified slightly. It will still recognize control 0 and stop echoing at that point, but the software will no longer look for CR as a terminator.

3. Any data read after a BREAK in TAPEMODE could cause the system to crash with an SF249. This has been resolved.

4. PTRIP was changed to recognize "OUT OF TAPE" prior to recognizing "NOT READY" from the paper tape punch.

i. SPOOLCOMS has been corrected to resolve a problem that occurred when attempting to display non-existent LDEV # when there were gaps in the LDEV # list.

about the HP 3000

j.   Enabling logging type 8 now changes job default INPRI to 8 as specified by the manual.

k.   PFAIL now checks LPDT for unused entries and ignores them upon recovery.

l.   MIOWAIT in the File System has been modified to avoid stack overflow.

m.   Item 1 mentioned below is an enhancement to SYSDUMP, #2 fixes existing problems.

   1.   SYSDUMP will now accept comments as part of the response to its questions. Everything from the first "<" to the last ">" or a carriage return will be ignored.

   2.   If in batch mode and the last word to be output is CR/LF, the CR/LF will be eliminated from the message.

n.   MMCORER has been corrected to pass CLABEL, not the segment displacement, back to the proper table when a segment is overlayed.

o.   PAUSE has been modified in PINT to correctly handle the overflow on converting seconds to milliseconds and converting REAL milliseconds to DOUBLE.

p.   GLOBALS has been modified in CROUTINE to correct some anomalies associated with the use of the E subqueue.

q.   MAM' DONE has been fixed to correct a recently reported problem which led to a system pause situation.

r.   CROUTINE has been modified to check only the low order 8 bits of the count of impeded EQ processes in order to determine if PRIUP should be called by a circularly scheduled process. It used to check all 16 bits which was in error.

s.   TICK (the routine that increments the time of day counter) has been moved to the CLOCKIO module in order to permit calling the TIMER intrinsic on the ICS without appearing to go backwards in time. Another problem resolved in TICK was to issue the correct control word to the clock after having executed a SIN instruction while the interrupt system was off. Under very special circumstances this would cause the system to lose actual time of day.

t.   FSPACE has been corrected to return CCG when it encounters a logical EOF on a mag tape file instead of CCE.

u.   The paper tape reader/punch routines have been modified as follows:

   1.   Reading Binary data will work now.

   2.   Punching a 0 length record will no longer cause the process to hang.

   3.   A tape low indication will now request the operator to mount a new tape.

v.   DPAN2 has been modified to avoid an integer overflow problem.

4.   ENHANCEMENTS TO MPEII A.00.05

a.   All Clock related routines have been gathered in a new module called CLOCKIO.

b.   PINT has been modified to kill all SONs of C.I. when "ABORTING".

c.   Starting with MIT NO.05/DATECODE 1646, the HP 7260A OMR (Optical Mark Reader) will be supported on the 3000 Series II.

FCARD Intrinsic —
The FCARD intrinsic allows you to control the operation of the 7260A OMR programmatically. This is achieved through passing a parameter (RECODE), with values corresponding to the function of FCARD desired, from your program to FCARD. FCARD returns to the program parameter, values which indicate the success or the cause of failure of execution the status of the 7260A, the file number of the 7260A/terminal file for which the function has been performed, and the number of columns read at the completion of a read request.

HP 7260A Application/ON-LINE Verifier Program —
The HP 7260A application/on-line verifier program provides three functions for the 7260A user through the FCARD intrinsic. This program:

   1.   Creates a disc file (ASCII) from cards (read by the 7260A) containing data and/or program code;

   2.   Creates a line printer listing from cards (read by the 7260A) containing data and/or program code; and

   3.   Verifies the 7260A/OMR operations while ON-LINE.

For further discussion of FCARD refer to MPE Intrinsics Reference Manual (HP Part No. 30000-90010) and for further discussion of the HP 7260A program refer to the 7260A Optical Mark Reader Operating Manual for 3000 Series II Computer System (HP Part No. 07260-90013).

5A. KNOWN PROBLEMS IN MPE C.00.14

a. Closing a tape file with no rewind is not implemented.

b. FSPACE spaces tape files by blocks rather than by records.

c. Chained SIOS on magnetic tape do not perform correctly, causing transfer of blocks larger than 4096 words to fail if the record format is variable or undefined.

d. The character ":" is treated as an EOF on $STDINX.

e. The commands: LISTACCT, LISTGROUP, and LISTUSER can lock the directory indefinitely if the output is written to magnetic tape and the tape is not ready.

f. If the FORMSG parameter of FOPEN begins on an odd byte boundary, the preceding byte is also printed.

g. Lower case :EOD is not recognized as an end-of-file on data accepting devices.

h. Issuing a :DEALLOCATE command for a non existent program file returns an ERR 217. The error should be ERR 217,52. The 52 is the file system error number returned from FCHECK.

i. Debug break points cannot be set in dynamically loaded procedures except by specifying the physical CST numbers.

j. When DPAN finds that the PCB table has been filled, it prints the erroneous messages "INVALID FIRST UN-ASSIGNED ENTRY" and "INVALID BACKWARD SUBQUEUE POINTER" even though there is no error in the PCB.

k. When the maximum number of open spoofles is not sufficient to handle all spooling requirements, spooled jobs may cause endless numbers of null list files to be generated. This bug manifests itself as multiple $STDLIST files for a single job, each producing only a header and trailer.

   ● If the line printer is spooled, this results in many null spoofles, each using four sectors of disc space.

● If the line printer is not spooled, these null spoofles will begin printing immediately (unless the printer is not ready). In this case, the system will crash due to an IOQ overflow.

● If an open spoofle is closed during this resource allocation loop, the job may execute normally. In this case, the last spoofle for $STDLIST will be the true job listing.

This bug can be overcome by increasing the maximum number of open spoofles. The recommended value is 20, but a more exact figure can be found by examining the usage of your system. Each initial allocation (FOPEN) of a spooled device uses one open spoofle. When the file is closed (FCLOSE), the spoofle becomes unopened.

For example:

A SESSION'S SINGLE ACCESS TO A SPOOLED LINE PRINTER REQUIRES ONE OPENED SPOOFLE: A SPOOLED JOB REQUIRES AT LEAST TWO, ONE FOR $STDIN AND ONE FOR STDLIST. EACH ADDITIONAL ACCESS TO A FILE OF DEVICE CLASS LP REQUIRES AN ADDITIONAL OPEN SPOOFLE.

One indication that the limit is being reached is allocation failures for spooled devices.

5B. KNOWN PROBLEMS IN MPEII A.00.05

System failure type:

a. SF130 — Memory manager loses a MTAB entry. If enough are lost a SF130 will occur.

b. SF311 — "Abort while critical".
Occurs when a code segment or memory links are getting destroyed.

c. SF30 — Under a very heavy load, one can get an SF30 while switching Log files.

FUNCTIONAL FAILURE TYPE:

1. A session with an outstanding READ cannot be aborted until reception of a carriage return.

2. A call to XARITRAP after a call with an illegal PLABEL returns garbage to OLDLABEL.

3. Several segmenter problems have been recorded recently. They are being worked on currently.

4. The FILESYSTEM seems to be building strange looking files for record sizes between 32640 and 32761.

about the HP 3000

2780/3780 EMULATOR, HP 30130B.02.00

This article describes the 2780/3780 Emulator, HP 30130 B.02.00 as incorporated in the 3000 pre-Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN THE 2780/3780 EMULATOR

Version B.01.03 of the 2780 Emulator corrects the HALT 14 caused by RJE. The HALT occurred when the line dropped before RJE sent a DLE EOT (BI-SYNC disconnect). The following RJE invocation caused an OPEN request to be linked into the previous CLOSE request. This is fatal when the CLOSE is serviced on the ICS. Modifications have been made to the driver to bypass sending a DLE EOT if the line has dropped. Modifications have been made to the monitor procedure to schedule linked requests in order to force processing off the ICS.

Version B.01.04 of the 2780 Emulator corrects the inability to receive data blocks of zero length ending with an ETX or ETB. The problem was resolved by a correction of the driver.

Version B.01.05 corrects a space compress error in the 3780 Emulator. Spaces were incorrectly compressed for exactly 40 spaces. The driver was also modified to drop DATA TERMINAL READY when DATA SET READY drops during a write continue.

The test file (T00T130B) was modified on MIT release data code 1623 to allow for changes made at the test host system.

Version B.02.00 corrects the following problems:

1. The driver now correctly receives transparent binary data at 4800 baud.

2. A bad entry in the ASCII/EBCDIC conversion table caused 3780 records with exactly 40 consecutive non-trailing blanks to be incorrectly compressed.

3. When a transmission error occurs during an #RJIN operation, a subsequent #RJEOD or #RJEND no longer attempts to transmit remaining buffered data.

4. 2780 punch data is now routed properly; it was previously routed to the list file.

5. Output to a file with a record length of 0 (e.g., $NULL) no longer results in an infinite loop.

6. Syntax errors now report the correct parameter number; previously, quotes, atsigns, and parantheses were added to the parameter number.

7. If the RJE command specifies the same disc file as the Command File and the Input File, or if an #RJIN command specifies a disc file that is the same as the Command File, command and input records are now processed correctly.

8. If an error occurs in loading a user input or output procedure, only the error that occurred is now reported. Previously, a PROCEDURE ERROR 39 was also sometimes reported.

9. The Emulator sometimes attempted to continue processing a command on which a syntactically incorrect file name, device, procedure name, or output count was specified. This has been corrected.

10. User punch and list procedures now work in offline operation (i.e., with SOURCE= specified). Previously, a syntax error was reported.

11. If an #RJIN with no file or device specification follows an #RJIN with a file or device specification, the second #RJIN now uses the correct record size. Previously, the record size of the prior file or device was used.

12. In an offline list or punch operation, if either the SOURCE file or the output file has a record length of 0, the Emulator no longer goes into an infinite loop.

13. In an offline list or punch operation, if an error occurs in reading the SOURCE file, the SOURCE file is now FCLOSE'd. Previously, it remained open, and if the error was repeated, a system failure could occur.

14. In an offline list operation, if the SOURCE file has carriage control, the carriage control is now used in the listing.

15. The Emulator manual does not document the fact that a BINARY input file (as determined by the FOPTION's for the file) specified on #RJIN defaults to transparent mode. To transmit a BINARY file in non-transparent mode, XPARENT=NO must be specified on the #RJIN command.

16. The Emulator previously recognized only the first component selection (routing) sequence received. This has been corrected.

17. The Emulator now allows up to 512 bytes to be transmitted in 3780 mode. It was allowing only 256.

18. When using the RIN feature on #RJLINE, the Emulator previously unlocked the RIN before closing the line; this could cause another user waiting on the RIN to fail on a line open. This has been corrected.

2780/3780 EMULATOR, HP 30130C.00.02

This article describes the 2780/3780 Emulator HP 30130C.00.02 as incorporated in the 3000 Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN THE 2780/3780 EMULATOR

Version C.00.01 corrects the following problem:

If "CONNECT = ANSWER" is not specified on #RJLINE, and a connection is not established within 15 minutes after the first transmit or receive operation, RJE will not terminate with a ****CS ERROR: 1,151 OR 2,151. Previously, RJE would wait indefinitely for the connection to be established.

Version C.00.02 corrects the following problems:

1. The Emulator previously recognized only the first component selection (routing) sequence received. This has been corrected.

2. The Emulator now allows up to 512 bytes to be transmitted in 3780 mode. It was allowing only 256.

3. If the Emulator is run from the MPE system console, echo is no longer suppressed during output operations.

4. When using the RIN feature on #RJLINE, the Emulator previously unlocked the RIN before closing the line: this could cause another user waiting on the RIN to fail on a line open. This has been corrected.

BASIC/3000, HP 32101B.00.05

This article describes BASIC/3000, HP 32101B.00.05 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN BASIC/3000

1. When reading from a variable-length file, the file LINPUT statement caused the same record to be re-read if a zero length record was encountered.

2. The construct "ELSE ON END . . . " caused a spurious "missing DOEND" error message and subsequently left the BASIC program in a bad state.

3. An IF END statement was not allowed as a THEN/ELSE clause, whereas an ON END statement was permitted.

4. A temporary file was not closed if the SPOOL command terminated with an error.

5. When its argument was a type-COMPLEX expression, the ATN function was evaluated in the wrong quadrant (reflected across the imaginary axis).

ENHANCEMENTS TO BASIC/3000

1. The VERSION command displays the BASIC Interpreter header. The command has no parameters. The VERSION command is useful for verifying the Interpreter version after having made several transactions in the Interpreter.

2. The WAIT command causes the BASIC Interpreter to be suspended (via the PAUSE intrinsic) for a specified period of time. The command has the following form:

    WAIT time1 [,time2]

where "time1" and "time2" have the following form:

    [[hours:] minutes:] seconds

"seconds" may be a floating-point value; it must be less than 60 if "minutes" is present. "Minutes" and "hours" must be integral values. "Minutes" must be less than 60 if "hours" is specified.

If only "time1" is specified, the Interpreter suspends for the indicated period of time. If "time2" is also specified, the Interpreter suspends for a random time period between "time1" and "time2". This command is useful for scheduling events during a benchmark.

```
*********************************************
*                                           *
*   THIS COMMAND CAN NOT BE TERMINATED      *
*              WITH CONTROL-Y               *
*                                           *
*********************************************
```

3. The Job Control Word (JCW) can be used to communicate a fatal error from a CALL'ed procedure to the Interpreter. After executing the CALL statement, if the JCW differs from its value before the CALL and it is non-zero, the Interpreter will terminate the BASIC program with the following message:

    name PROCEDURE ERROR number IN LINE . . .

where "name" is the name of the procedure in the CALL statement, and "number" is the value of the JCW. Note that the Interpreter examines only the lower 15 bits of the JCW; thus, a non-zero value must be placed there to effect an error termination. The Interpreter resets that portion of the JCW to zero after reporting the error. Setting the JCW to zero or not altering the JCW at all will be interpreted as a "no error" situation. The JCW can be modified by calling the SETJCW intrinsic.

about the HP 3000

If the CALL'ed procedure sets the sign bit of the JCW, the job will be aborted (with an MPE error #2) only after the Interpreter is exited. Placing a :CONTINUE command before the :BASIC command will prevent the job from terminating abnormally.

KNOWN PROBLEM IN BASIC/3000

1. The interpreter aborts with a stack underflow when control-Y is typed in certain circumstances. This occurs most often when printing the FREQ table. The problem may also arise in some cases when INVOKing or using the ABORT, CALLS or FILES commands in BREAK-mode. [BR #1396]

   Work-around (for BREAK-mode commands): Type control-Y and set a breakpoint at the next statement to be executed. Then enter the GO or RESUME command. When you break at the next statement, it will then be safe to use any BREAK-mode commands.

BASICOMP/3000, HP 32103B.00.02

This article describes BASICOMP/3000 HP 32103B.00.02 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN BASICOMP/3000

1. A compiled program would abort with a spurious "invalid string" error when the last element of the statement (eg., LET or PRINT statement) was a simple string reference without substring designators and the following statement was the asterisk-form of the CALL statement.

2. A compiled program would abort with a spurious "invalid string" error when a string reference included a substring designator of the following form: S$(X,255), where "X" is some non-constant expression and the last-character designator is the constant 255.

3. Bad code was generated when an INPUT statement was followed by a file MAT PRINT statement under one of two conditions:

   a) no input item list was specified on INPUT statement, or

   b) the last item in input item list was a prompt string.

4. Anomalous results occurred (eg., improper execution of FOR loops) when the last item of a PRINT statement was a numeric expression.

5. Compile-time errors and abnormal termination of compiled programs caused the Job Control Word (JCW) to be set to %100000, destroying any value that may have been set in previous job steps. The sign bit of the JCW is now set without altering the other bits.

6. FOR loops improperly executed when an UPDATE statement within the loop specified a type-LONG or type-COMPLEX expression (rather than just a simple variable) for the new value of the data item.

7. Compiled programs would intermittently abort with a "stack underflow" error during the execution of a CALL statement.

8. Compiled programs would abort with a "bounds violation" error when redimensioning a string array to size larger than its current logical size.

9. The ADVANCE statement would fail to position correctly in the data file when a backward movement was indicated (ie., a negative item count) and the desired item was not in the current or previous record.

10. Reads and prints to a file containing variable-length records caused a premature end-of-file error if the record number exceeded the maximum number of physical records ("LIMIT" in a :LISTF) but did not exceed the number of logical records ("EOF" in a :LISTF).

11. Execution of a file PRINT statement ending with the keyword "END" (eg., PRINT #f; END) failed to set an end-of-file in the last record of a BASIC formatted data file.

12. $NULL was always treated as binary file, causing the ADVANCE, file LINPUT and UPDATE statements to terminate with an "incorrect file-type" error. These statements now treat $NULL as a file of an appropriate type and act accordingly.

13. When opening the same file a second time (eg., FILES A,A), the second file was permitted only read access.

Errata

1. The following problem was fixed in version B.0.1 but omitted from the list of corrected problems. A compiled program would abort with a spurious "invalid file number" error when attempting to execute a direct file I/O statement which specified a type-LONG or type-COMPLEX record number.

OUTSTANDING PROBLEMS IN BASICOMP/3000

1. When the base is type-REAL and the power is a type-LONG constant representable as an integer (eg., 9∗∗(2L0)), single-precision rather than double-precision arithmetic is performed SPR #2007.

   Work-around: replace type-LONG constant power with a variable.

2. The unary-minus operator preceding a constant is not always handled correctly. This causes the following two incorrect results (where "x" represents a constant and "y" represents any variable, constant or expression):

   a) "−x MOD y" is evaluated as "(−x) MOD y" instead of "−(x MOD y)"

   b) "−x∗∗y" is evaluated as "(−x)∗∗y" instead of "−(x∗∗y)". SPR #2008

   Work-around: Fully parenthesize expression.

3. Incorrect code is generated for I/O FOR-loops with a constant negative one STEP size. SPR #6006.

   Work-around: Replace the constant STEP size with a variable.

4. Incorrect code is generated for a FOR-loop which encloses both an ONEND statement with a destination within the loop and a GOTO statement with a destination outside the loop. This situation will cause spurious run-time aborts if an end-of-file is detected while inside the FOR-loop.

   Work-around: Place a superfluous GOTO statement outside the FOR-loop with a destination inside the loop. The GOTO statement itself is not intended to be executed.

5. Lower-case characters are not recognized as format specifications in PRINT USING format string expressions.

6. A bounds violation or other anomalous results occur when a user-defined function is used within a subscript expression on the left-hand side of a LET statement. For example:

   X(FNA(Y)) = 10

Work-around: Eliminate the reference to the user-defined function in the subscript expression by evaluating it in a preceding statement. For example:

   Z=FNA(Y) Y(Z)=10

COBOL/3000, HP 32213B.02.03

This article describes COBOL/3000, HP 32213B.02.03 as incorporated in the 3000 Pre Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN COBOL/3000

1. An EXAMINE statement with an indexed data item caused an ERROR 129 at compile time.

2. A USL file specified as the LIST file produced an error message and destroyed the contents of the USL file. Now, the USL file remains intact.

3. A grammatical error in the Data Division caused subsequent PICTURE clauses with parentheses to be flagged as errors also.

4. An End-of-File on the LIST file sometimes produced the wrong error message.

5. Using a DISPLAY or COMP-3 item as a subscript when comparing a DISPLAY item with a COMP-3 item did not work properly.

6. Moving a COMP-3 item to an unsigned COMP-3 item of different size generated a HEX C instead of a HEX F as the sign.

7. Several errors in complex COMPUTE statements were corrected.

8. An Error 212 (Code Segment Exceeds 16K) now gives the paragraph name in which the overflow first occurred.

9. Compiling a program or subprogram into a USL file in which the same program unit had previously been purged did not always work.

10. A compound conditional using class names (such as, IF A IS NOT NUMERIC OR B IS NOT NUMERIC) did not always work properly.

11. A MULTIPLY statement with two operands such that the product could exceed 28 digits (before truncating for storing into the receiving field) did not work properly.

ENHANCEMENTS TO COBOL/3000

1. Subprograms no longer have to begin at PB+0 (thus, you can combine COBOL subprograms using the Segmenter).

2. Several constructs now produce more efficient object code; for example, character comparisons (of the same length), literal moves, and COMP moves (of the same size and characteristics).

PROBLEMS CORRECTED IN THE COMPILER LIBRARY FOR COBOL HP 32211C.04.04

1. Several routines aborted with Bounds Violations when working with data stacks greater than 32K bytes. The routines affected were: C'ACCEPTCONSOLE, C'LIT'FIG'MOVE, C'MORELABELS, C'OPEN, C'OPEN', C'READ, C'READD, C'SEEK, C'SORTERR, C'WRITE, C'WRITE', C'WRITED, C'WRITESEQ', and C'WRITESEQ.

2. An Error 711 (Invalid ASCII Digit) in a CVDA (Convert ASCII to Decimal) instruction was not handled properly.

KNOWN PROBLEMS IN COBOL/3000

1. Putting a non-dynamic subprogram into an RL produces an Invalid Program File at run time.

2. If the last unit in a USL file (that is, the one at the end of a LISTUSL listing) is a non-dynamic subprogram, moving it to another segment can cause an Invalid Program File at run time.

COBOL/3000, HP 32213C.01.01

This article describes COBOL/3000, HP 32213C.01.01 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN COBOL/3000

1. A GO TO statement to the first paragraph in a subprogram did not work properly.

2. A RENAMES clause was erroneously being detected as an error if there were any OCCURS clauses in the record. Now only the following OCCURS clause conditions are errors in RENAMES clauses:

   a. The item being renamed is subordinate to an OCCURS clause.

   b. The item being renamed contains an OCCURS clause.

   c. The item being renamed has a subordinate item containing an OCCURS clause.

3. An illegal RENAMES clause sometimes caused the compiler to loop or abort.

4. Compiling into a non-empty USL file which was built with more than 2040 records sometimes caused the USL file to be initialized even though $CONTROL USLINIT was not specified.

5. Compiling a subprogram containing secondary entry points with parameters into an non-empty USL file did not inactivate previous entry points with the same name.

6. Compiling programs into a non-empty USL file sometimes caused the compiler to loop. This problem usually occurred only when secondary entry points were present in the program being compiled.

7. The COBTEMP file filled up during compilation, thereby aborting the compiler.

8. An EXAMINE statement with an indexed data item caused an erroneous ERROR 129 at compile time.

9. A USL file specified as the LIST file produced an error message and destroyed the contents of the USL file. Now, the USL file remains intact.

10. A grammatical error in the Data Division causes subsequent PICTURE clauses with parentheses to be flagged as errors also.

11. An End-of-File on the LIST file sometimes produced the wrong error message.

12. Using a DISPLAY or COMP-3 item as a subscript when comparing a DISPLAY item with a COMP-3 item did not work properly.

13. Moving a COMP-3 item to an unsigned COMP-3 item of different size generated a HEX C instead of a HEX F as the sign.

14. Several errors in complex COMPUTE statements were corrected.

15. An Error 212 (Code Segment Exceeds 16K) now gives the paragraph name in which the overflow first occurred.

16. Compiling a program or subprogram into a USL file in which the same program unit had previously been purged did not always work.

17. A compound conditional using class names (such as, IF A IS NOT NUMERIC OR B IS NOT NUMERIC) did not always work properly.

18. A MULTIPLY statement with two operands such that the product could exceed 28 digits (before truncating for storing into the receiving field) did not work properly.

19. The compiler would sometimes abort with a Bounds Violation when trying to produce a symbol map for a program with errors in it.

ENHANCEMENTS TO COBOL/3000

1. Subprograms no longer have to begin at PB+0 (thus, you can combine COBOL subprograms using the Segmenter).

2. Several constructs now produce more efficient object code; for example, character comparisons (of the same length), literal moves, and COMP moves (of the same size and characteristics).

PROBLEMS CORRECTED IN THE COMPILER LIBRARY FOR COBOL, HP 32211C.04.04

1. Several routines aborted with Bounds Violations when working with data stacks greater than 32K bytes. The routines affected were: C'ACCEPTCONSOLE, C'LIT'FIG'MOVE, C'MORELABELS, C'OPEN, C'OPEN', C'READ, C'READD, C'SEEK, C'SORTERR, C'WRITE, C'WRITE', C'WRITED, C'WRITESEQ', and C'WRITESEQ.

2. An Error 711 (Invalid ASCII Digit) in a CVDA (Convert ASCII to Decimal) instruction was not handled properly.

KNOWN PROBLEMS IN COBOL/3000

1. Putting a non-dynamic subprogram into an RL produces an Invalid Program File at run time.

2. If the last unit in a USL file (that is, the one at the end of a LISTUSL listing) is a non-dynamic subprogram, moving it to another segment can cause an Invalid Program File at run time.

CROSSLOADER, HP 32226A.02.00

This article describes the Crossloader, HP 32226.02.00 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1636.

MODIFICATIONS/ENHANCEMENT TO THE CROSSLOADER

1. The output file (created as a result of the OUTPUT command) now has fixed length records. This results in better performance when doing a large relocation with the 'LINKS ON' option because the FPOINT intrinsic can be used to position the file rather than rewinding and reading sequentially.

2. Files specified in a SEARCH are now verified to be fixed or undefined format. Prior to this a SEARCH on a variable record length would lose entry points.

3. A new command (LOCK SYS) has been added to freeze the Loader Symbol Table after an operating system and library has been generated. Used in conjunction with the 'CLEAR PROG' command.

4. A new command (CLEAR PROG) has been added to allow the Loader Symbol Table to be cleared of entries which are entered as a result of a relocation and search. This will allow the Cross RTE-C Generator to handle Type 7 subroutines which are appended to each program which calls them.

5. A new format will be used when displaying keyboards.

FORTRAN/3000, HP 32102B.00.05

This article describes FORTRAN/3000 HP 32102B.00.05 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN FORTRAN/3000

1. Secondary entry points were not always being ceased correctly in USL's. This problem was manifested on the PREP of such an invalid USL (SPR 1224)

2. A problem was discovered in using variable length CHARACTER variables as parameters to subroutines and functions which contain secondary entry points. The problem causes very odd program operation, usually terminating with almost any kind of abort. (SPR 1227)

3. A logical expression using the .EQ. operator on character variables or constants of different lengths always returned .TRUE. instead of .FALSE.

4. There was a problem with the cross reference facility when the value of the function being compiled was passed as a parameter to another function or subroutine. For example,

        FUNCTION A(B)
        . . .
        CALL C(A)
    (SPR 1100)

5. $CONTROL CHECK= 0,1, or 2 before the main program caused the compiler to abort (in logical CST 4) or to loop. The failure to check for this user error caused the compiler to attempt to generate an incorrect USL entry, which ultimately caused the compiler fault. Incidentally, a common misconception about the CHECK= capability is that it turns off checking on all function and subroutine calls from a particular program unit. As noted in the FORTRAN manual, changing the CHECK level of a subroutine or function merely means that calls TO that program unit need not agree with the declaration of that program unit. It does NOT affect the checking of calls FROM that program unit, which is still set to the maximum of 3. In other words, a subroutine or function may permit calls to itself with less than maximum checking. It may not call another subroutine or function with a check level less than that other subroutine or function explicitly or implicitly permits. (SPR 1279)

6. The compiler failed to detect a non-dimensioned variable used with a subscript in a main program. (SPR 1236)

QUERY/3000, HP 32216A.03.02

This article describes QUERY/3000, HP 32216A.03.02 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEMS CORRECTED IN QUERY/3000

1. The "LIST" setname FOR relation" command often failed to work correctly, responding that the 'setname' was illegal when in fact it was perfectly valid. This condition has been corrected.

2. Previously a FIND CHAIN command followed by a REPORT command would generate the error "BAD DATA SET OR DATA ITEM REFERENCE (SUB SYSTEM ERROR)" if no records of the FIND CHAIN search (other than the indexing masters) had qualified. As a REPORT command only operates on a single record type (with the exception of REPORT ALL), the same sequence now generates the message "NO RECORDS TO REPORT" to indicate that no detail records qualified, even though one or more master records used in the search qualified.

3. A numeric edit mask of all 9's used to suppress leading zeros. It now functions properly, such that each "9" in the edit mask is replaced with a decimal digit from the data item value, zero or otherwise.

RTE PROGRAMMABLE CONTROLLER, HP 30301B.00.01

This article describes the RTE Programmable Controller HP 30301B.00.01 as incorporated in the Series II MIT tape date coded 1636.

MODIFICATIONS TO THE RTE PROGRAMMABLE CONTROLLER

The Interrupt Table was not being handled correctly when a power fail routine was included as any entry but the last one.

Type 7 utility subroutines were not being appended to each program which called them. They were being loaded only once. This change requires the use of XL2100.02.0 or later. Earlier versions of the Crossloader will abort with an invalid command message.

The procedure DNLDUSER has been modified to transmit only the significant words in an absolute record rather than the actual number of words in the record. This is done by using the length in the left 8 bits of the first word and adding three.

This version of P.C. has a new driver for the Series II.

SORT/3000, HP 32214B.01.02

This article describes SORT/3000, HP 32214.01.02 as incorporated in the 3000 Pre Series II and Series II MIT tape date coded 1646.

PROBLEM CORRECTED IN SORT/3000

1. Reaching the end of tape on a MERGE output file now requests another tape instead of aborting.

documentation

The following tables list currently available customer manuals for HP 3000 products. This list supersedes the list in the last issue of the **Communicator.**

The most recent changes to the tables are indicated for easy reference. Prices are subject to change without notice.

Copies of manuals and updates can be obtained from your local Sales and Service office. The address and telephone number of the office nearest to you are listed in the back of all customer manuals.

Update packages are free of charge. If you require an update package complete the Update Order Form in the back of the **Communicator** and mail the form to:

> Software/Publications Distribution
> 5303 Stevens Creek Blvd.
> Santa Clara, Ca 95050

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manual(s) you need on the Corporate Parts Center form supplied at the back of the **Communicator.**

A few words about documentation terms:

New   A new manual refers only to the first printing of a manual. When first printed, a manual is assigned a part number.

Revised   A revised manual is a printing of an existing manual which incorporates new and/or changed information in its contents. For example, a manual is revised when an update package is incorporated into the manual: the manual gets a new print date and the update package disappears. Note that a revision to a manual effectively obsoletes the previous version of the manual.

Update   An update package is a supplement to an existing manual which contains new and/or changed information. Updates are issued when information must get to customers, yet it is inappropriate to issue a revised manual. An update has no part number, it is automatically included when you order the manual with which it is associated.

MPE/3000 MANUALS

PRE-SERIES II		MANUAL TITLE	PART NUMBER	PRICE	SERIES II	
PUBLICATION DATE	UPDATE				PUBLICATION DATE	UPDATE
		Console Operator's Guide	30000-90013	$ 7.00	6/76	10/76
1/75	11/76	Console Operator's Guide, 32000C MPE/3000	32000-90004	7.00		
		Error Messages and Recovery Manual	30000-90015	17.00	6/76	
		General Information Manual	30000-90008	6.50	10/76*R	
11/73		General Information Manual, Multiprogramming Executive	03000-90096	4.00		
		HP 3000 CX to HP 3000 Series II Program Conversion Guide	30000-90046	4.00	6/76	
		Index to MPE Reference Documents	30000-90045	5.50	6/76	
1/75	6/76	MPE/3000 Reference Manual, 32000C	32000-90002	19.50		
		MPE Commands Reference Manual	30000-90009	12.50	6/76	
		MPE Intrinsics Reference Manual	30000-90010	15.00	6/76	10/76
		MPE Segmenter Reference Manual	30000-90011	4.00	6/76	
		MPE Debug/Stack Dump Reference Manual	30000-90012	7.50	9/76	9/76[†]
		MPE System Utilities Reference Manual	30000-90044	5.00	6/76	
10/75		MPE/3000 Operating System, System Utilities Manual	32000-90008	3.00		
		Software Pocket Guide	30000-90049	3.50	6/76	
7/75		Software Pocket Guide, HP 3000	03000-90126	3.50		
		System Manager/System Supervisor Manual	30000-90014	10.00	6/76	9/76
10/75		System Manager/System Supervisor Manual, 32000C MPE/3000	32000-90006	14.00		
6/75		Using the HP 3000: A Guide for the Terminal User	03000-90121	7.50	6/75	

*R = Revised Manual

about the HP 3000

LANGUAGE MANUALS

| PRE-SERIES II | | | PART NUMBER | PRICE | SERIES II | |
PUBLICATION DATE	UPDATE				PUBLICATION DATE	UPDATE
		APL 3000 Reference Manual	32105-90002	$12.00	11/76*N	
		APL 3000 Pocket Guide	32105-90003	1.50	11/76*N	
		BASIC Interpreter Reference Manual	30000-90026	11.50	6/76	
7/75		BASIC/3000 Interpreter Reference Manual	03000-90008	10.00		
9/74		BASIC/3000 Interpreter Pocket Guide	03000-90050	2.50		
11/74		BASIC/3000 Compiler Reference Manual	32103-90001	3.50	11/74	6/76
11/72		BASIC for Beginners	03000-90025	5.50	11/72	
7/75		COBOL/3000 Reference Manual	32213-90001	12.50	7/75	6/76
5/76		Cross Assembler for 2100 Computers Reference and Application Manual	03000-90047	12.00	5/76	
		FORTRAN Reference Manual	30000-90040	9.50	6/76	
3/76		FORTRAN/3000 Reference Manual	32102-90001	13.50		
11/76*N		FORTRAN/3000 Pocket Guide	32102-90002	1.50	11/76*N	
2/75	8/76	RPG/3000 Compiler Reference and Application Manual	32104-90001	16.50	2/75	8/76
4/75		RPG Listing Analyzer	32104-90003	0.50	4/75	
11/73		SPL/3000 Reference Manual	03000-90002	7.50		
11/73	3/75	SPL/3000 Textbook	03000-90003	13.00		
		SPL Pocket Guide	32100-90001	2.00	11/76*N	
		System Programming Language Reference Manual	30000-90024	15.00	9/76	10/76
		System Programming Language Textbook	30000-90025	11.00	6/76	10/76

†There are instances where a new edition of the manual and an update are both shown with the same date. The new edition incorporates all changes made in the update. If you already have the manual, order the update only; otherwise order the new edition.

*N = New Manual (Refer to the bulletin section.)

ADDITIONAL MANUALS

| PRE-SERIES II | | MANUAL TITLE | PART NUMBER | PRICE | SERIES II | |
PUBLICATION DATE	UPDATE				PUBLICATION DATE	UPDATE
12/74	2/76	2780/3780 Emulator Reference Manual	30000-90047	$ 8.50	6/76	
		2780/3780 Emulator Subsystem Reference and Application Manual	30130-90001	10.00		
		Compiler Library Reference Manual	30000-90028	12.00	11/76*R	
2/76		Compiler Library Reference Manual, HP 3000	03000-90009	16.50		
		Data Entry Library Manual	30000-90050	6.50	6/76	
8/75		EDIT/3000 Reference Manual	03000-90012	7.50	8/75	6/76
6/76		FCOPY Reference Manual	03000-90064	6.00	6/76	
6/76		HP 2894A Card Reader Punch Operating and Programming Manual	30119-90009	7.00	6/76	
10/74		HP 3000 Cross Loader for HP 2100 Computer	03000-90107	11.00	10/74	6/76
2/75	5/75	IBM 1130/1800 to HP 3000 FORTRAN Conversion Guide	36995-90013	6.00		
12/75		IBM System/3 to HP 3000 Conversion Guide	32104-90004	7.50		
		IMAGE Data Base Management System Reference Manual	30000-90041	4.50	6/76	

ADDITIONAL MANUALS (Continued)

PRE-SERIES II		MANUAL TITLE	PART NUMBER	PRICE	SERIES II	
PUBLICATION DATE	UPDATE				PUBLICATION DATE	UPDATE
3/76		IMAGE/3000 Reference Manual	32215-90001	7.00		
		Instruction Decoding Pocket Guide	30000-90057	1.00	6/76	
		Line Printer Operating and Programming Reference Manual	30209-90008	6.00	6/76	
		Machine Instruction Set Reference Manual	30000-90022	7.00	6/76	
		Programmable Controller Reference and Application Manual	30000-90066	6.00	6/76	10/76
2/75	7/76	Programmable Controller Reference and Application Manual	30300-90002	12.00		
		QUERY Reference Manual	30000-90042	6.50	6/76	
3/76		QUERY/3000 Reference Manual	32216-90001	7.00		
		Real-Time Programmable Controller Reference and Application Manual	30000-90067	7.50	6/76	
2/75	7/76	Real-Time Programmable Controller Reference and Application Manual	30301-90002	9.50		
		Scientific Library Reference Manual	30000-90027	5.00	6/76	
7/75		Scientific Library Reference Manual, HP 3000	03000-90010	7.00		
		Site Preparation Manual	30000-90016	6.00	6/76	10/76
		Site Planning Workbook, HP Computer System	30000-90017	10.00	6/76	
8/76		Sort/3000 Reference Manual	32214-90001	6.50	8/76	
7/75		Student Assignment System Reference Manual	32901-90001	10.00	7/75	8/76
7/75		Student Assignment System — Technical Manual	32901-90005	13.00	7/75	
9/74		Student Information System Reference Manual	32900-90001	18.00	9/74	8/76
9/74		Student Information System -- System Overview	32900-90002	7.00	9/74	
3/75		Student Information System -- Technical Manual	32900-90005	18.50	3/75	
		Systems Reference Manual	30000-90020	9.50	6/76	11/76
9/73		Systems Reference Manual, HP 3000 Computer	03000-90019	14.00		
6/76		Trace/3000 Reference Manual	03000-90015	7.00	6/76	

*R = Revised Manual

training schedule

about the HP 3000

The schedule for customer training courses on General Systems Division products is outlined below, and in the 2000 section of this publication. Included here are HP 3000 software courses offered in the U.S. and in Europe for the period January 1977 through March 1977. You can also obtain a copy of the schedule from your local HP sales office. A European course schedule is available through the sales offices in Europe; a U.S. schedule through U.S. sales offices.

Registration

Requests for enrollment in any of the courses should be made through your local HP Sales Office. Your Sales Representative will supply the Training Registrar at the appropriate location with the course number, dates, and requested motel reservations. Enrollments are acknowledged by a written confirmation indicating the training course, time of class, location and accommodations reserved.

GENERAL SYSTEMS DIVISION COURSE SCHEDULE (U.S.)

January — March 1977					
				Course Dates and Training Center Location	
NUMBER	**COURSE TITLE**	**LENGTH**	**PRICE**	**WESTERN TECHNICAL CENTER**	**EASTERN TECHNICAL CENTER**
22801A	3000 Series II, A Comprehensive Introduction	5 days	$500	Jan. 3, 1977 Jan. 17, 1977 Jan. 31, 1977 Feb. 14, 1977 Feb. 28, 1977 Mar. 14, 1977 Mar. 28, 1977	Jan. 3, 1977 Jan. 24, 1977 Feb. 7, 1977 Feb. 28, 1977 Mar. 21, 1977
22802A	3000 Series II, System Management and Operation	4 days	400	Jan. 10, 1977 Jan. 24, 1977 Feb. 7, 1977 Feb. 21, 1977 Mar. 7, 1977 Mar. 21, 1977	Jan. 10, 1977 Jan. 31, 1977 Feb. 14, 1977 Mar. 7, 1977 Mar. 28, 1977
22956A	3000 IMAGE	5 days	500	Jan. 17, 1977 Jan. 31, 1977 Feb. 28, 1977 Mar. 14, 1977	Jan. 17, 1977 Mar. 14, 1977
22957A	3000 COBOL, Audio Self Study	30 hrs.	325	These courses can be ordered using the Direct Mail Order form in the back of the **Communicator**.	
22958A	3000 BASIC, Audio Self Study	30 hrs.	325		

Accommodations

Students provide their own transportation, meals, and lodging. The Training Registrar will be pleased to assist in securing motel reservations at the time your Sales Office requests a registration.

Cancellations

In the event you are unable to attend a class for which you are registered, please notify your HP Sales Office immediately. To avoid paying for a reservation which you do not use, we must receive notification of your cancellation no later than 10 working days before the class begins.

GENERAL SYSTEMS DIVISION COURSE SCHEDULE (EUROPE)

COURSE NUMBER	COURSE TITLE	LENGTH	FRANKFURT/ BÖEBLINGEN* (GERMAN)	WINNERISH (ENGLISH)	ORSAY (FRENCH)	MILAN (ITALIAN)	STOCKHOLM (ENGLISH)
22801A	3000 Series II, A Comprehensive Introduction	5 days	Jan 10 Mar 21 May 23		Jan 10	Apr 18	Jan 31 Apr 25
22802A	3000 Series II, System Management and Operation	4 days	Jan 24 Apr 4	Jan 24 Mar 28 May 23	Feb 28 Apr 18	May 2	Feb 7 May 2
22956A	3000 IMAGE	5 days	Feb 7 Apr 18	Feb 21 Apr 25 Jun 20	Jun 6 Aug 29	May 16	Feb 14

January — April 1977

Technical Center Addresses

UNITED STATES

Eastern Technical Center	Western Technical Center
4 Choke Cherry Road Rockville, Maryland 20850	5303 Stevens Creek Blvd. Santa Clara, California 95050

EUROPE

Winnersh

King Street Lane
Winnersh, Wokingham
Berks RG11 5 AR
Tel: Wokingham 784774
Cable: Hewpie London
Tele: 847178 9

Orsay

Quartier de Courtaboeuf
Boite postale No. 6
F-91401-Orsay
France
Tel: (1) 907 78 25
Cable: HEWPACK Orsay
Telex: 60048

Frankfurt

Vertriebzentrale Frankfurt
Berner Strasse 117
Postale 560 140
D-6000 Frankfurt 56
Tel: 0611 5004
Telex: (841) 04-13249, 04-13081
Cable: HEWPACKSA Frankfurt

Böeblingen*

Herrenberger Strasse 130
7030 Böeblingen, W. Germany
Telefon 07031/667-540
Telex 7265739

Milan

Via Amerigo Vespucci, 2
1-20124 Milan
Tel: (2) 62 51
Cable: HEWPACKIT Milano
Telex: 32046

Stockholm

Enighetsvagen 1-3, Fack
S-162 20 Bromma 20
Tel: (08) 730 05 50
Cable: MEASUREMENTS
Stockholm Telex: 10721

Effective March 1, 1977 classes will be held in Böeblingen.

NOT TO BE USED FOR ORDERING COMMUNICATOR SUBSCRIPTIONS

HEWLETT **hp** *PACKARD*

CORPORATE PARTS CENTER
Direct Mail
Parts and Supplies Order Form

SHIP TO:

NAME _____

COMPANY _____

STREET _____

CITY _____ STATE _____

CUSTOMER
REFERENCE # _____

TAXABLE*? _____

ZIP CODE _____

Item No.	Check Digit	Part No.	Qty.	Description	List Price Each		Extended Total	

Special Instructions	Sub-total		
	Your State & Local Sales Taxes*		
	Handling Charge	1	50
	TOTAL		

*Tax is verified by computer according to your ZIP CODE. If no sales tax is added, your state exemption number must be provided: # _____ . If not, your order may have to be returned.

Check or Money Order, made payable to Hewlett-Packard Company, must accompany order.

When completed, please mail this form with payment to:

HEWLETT-PACKARD COMPANY
Mail Order Department   Phone: (415) 968-9200
P.O. Drawer #20
Mountain View, CA 94043

Most orders are shipped within 24 hours of receipt. Shipments to California, Oregon and Washington will be made via UPS. Other shipments will be sent Air Parcel Post, with the exception that shipments over 25 pounds will be made via truck. No Direct Mail Order can be shipped outside the U.S.

NOTE: This order form cannot be used for ordering manuals. It is for replacement pages only.

HEWLETT **hp** *PACKARD*

SOFTWARE/PUBLICATIONS DISTRIBUTION
ORDER FORM

UPDATES TO 3000 AND 2000
ACCESS LEVEL MANUALS ONLY

SHIP TO:

NAME _____

COMPANY _____

STREET _____

CITY _____ STATE _____ ZIP CODE _____

MANUAL NAME	PART NUMBER	QUANTITY

When completed, please   *There is no charge for manual updates.*
mail this form to:

HEWLETT-PACKARD
SOFTWARE/PUBLICATIONS DISTRIBUTION
5303 Stevens Creek Blvd.
Santa Clara, CA   95050

Although every effort is made to insure 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.