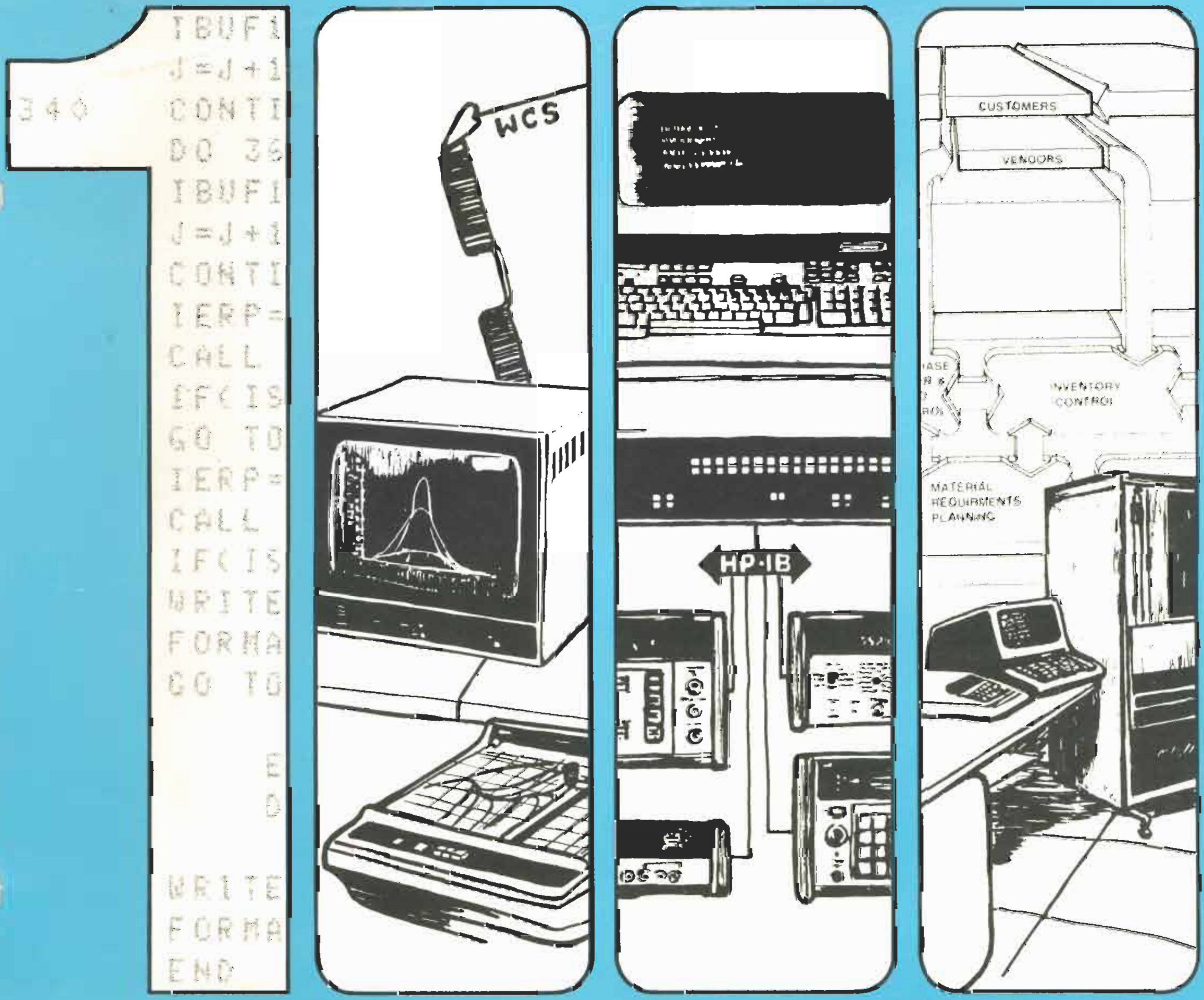
HEWLETT hp PACKARD

Computer Systems COMMUNICATOR



1979

Volume III Number 1

HEWLETT-PACKARD
COMPUTER SYSTEMS

Volume III
Number 1

COMMUNICATOR/1000

Feature Articles

Departments

EDITOR'S DESK

ABOUT THIS ISSUE

This issue of COMMUNICATOR/1000 inaugurates a new feature article section, DATA COMMUNICATIONS. The section contains two important tutorial articles. The first, "A MINICOMPUTER-BASED RESOURCE SHARING DATAGRAM NETWORK", describes the technical capabilities of Distributed Systems/1000 networking software and the functions performed by each layer of software. Have you ever wondered what really happens when a user program requests access to a resource on another computer in the DS network? The article traces a network transaction from initiation to completion to answer this question in detail. It is reprinted from a paper presented at a May, 1978 symposium on Trends and Applications: Distributed Processing sponsored by the National Bureau of Standards and the IEEE Computer Society. The author, Robert R. Shatzer, has been the project leader for DS/1000 networking software. The second article in the new section, "UNDERSTANDING RJE/1000", outlines the capabilities provided by remote job entry software and then explores the now pervasive Bisync protocol upon which all IBM 2780's (and their emulators) are built. Of particular interest is the description of the communications line TRACE capability added to the 1840 (October, 1978) Revision of the product. The author, Robert Gudz, is a member of the HP 1000 R&D Lab.

Data communications is an important part of our business and therefore of our customers' businesses. HP has been an innovator in this field for a long time. In 1973, HP delivered the world's first standard product computer networking software package; it was designed to meet the resource sharing and information management needs of our instrumentation customers. It has been enhanced over several generations to become the DS/1000 product of today — a product with a uniquely powerful and flexible nodal network architecture. Today, there are over 1500 Distributed Systems network nodes installed. More recently, HP has innovated in the data communications field with a microprocessor-based Multipoint Terminal Interface on a single I/O card.

The new DATA COMMUNICATIONS section has been added to the COMMUNICATOR to foster a healthy and vigorous interchange of technical ideas and accomplishments in this dynamic and rapidly growing field. The Editor hopes readers will contribute technical articles relating to computer-to-computer and terminal-to-computer communications.

The issue also has three excellent articles in the OPERATING SYSTEMS section. John Blommers, from the Defense Research Establishment Pacific in Canada, has written a valuable article for RTE-II users who are writing and debugging their own drivers. He presents a subroutine that allows the driver to be loaded on-line with the user program into a memory partition. System regeneration to test a new version of the driver is therefore not required. Bob Sauers, HP Rockville, has contributed a very interesting article that clears up some common misunderstandings concerning the real meaning of disc track sparing. Sparing on 7900 Discs is contrasted with sparing on 7906 and 7920 Discs. The three ways spare tracks can be activated in RTE are also discussed. Alan Housley, HP Data Systems Division, presents an invaluable program which finds all files with extents on a particular disc LU, combines file extensions into larger, continuous files, and clears the extents.

In the OPERATIONS MANAGEMENT section Erik M. Best of General Electric Broadcasting has presented an excellent top-down approach to the process of "MAINTAINING A VALID DATA BASE" given that the system has gone down. He discusses how to detect a corrupt data base and how to reconstruct it quickly.

Given such a strong article line-up, it is no easy task to choose the best articles and therefore the winners of HP-32E Calculators. However, three impartial judges have chosen the following articles to be the best in terms of technical content, completeness of the subject covered and general interest to COMMUNICATOR readers:

For a Feature Article by a Customer	**RELOCATABLE DRIVERS FOR RTE-II** John Blommers
For a Feature Article by an HP Employee in the Field	**WHAT IS DISC TRACK SPARING?** Bob Sauers

EDITOR'S DESK

For a Feature Article by an HP Data Systems Employee not in the Technical Marketing Dept.	**A MINICOMPUTER-BASED RESOURCE SHARING DATAGRAM NETWORK** Robert R. Shatzer

Hearty CONGRATULATIONS to the WINNERS!!

It's hoped you'll find this COMMUNICATOR helpful and enjoyable.

William H. Stevens
Editor, Vol. III, No. 1

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

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

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

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

WIN AN HP-32E CALCULATOR!

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

1. HP 1000 Customers;

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

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

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

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

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

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

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

   OPERATING SYSTEMS
   DATA COMMUNICATIONS
   INSTRUMENTATION
   COMPUTATION
   OPERATIONS MANAGEMENT

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

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

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

A SPECIAL DEAL IN THE OEM CORNER

When an HP 1000 OEM writes a Feature Article that is not only technically detailed and insightful but also application-oriented as opposed to theoretical, then that OEM may ask that the article be included in THE OEM CORNER. A Feature Article included in THE OEM CORNER may contain up to 150 words of pure product description as well as a picture or illustration of the OEM'S product or its unique contribution. HP's objective is twofold: (1) to promote awareness of the capabilities HP 1000 OEMs' products among all HP 1000 users; and, (2) to publish an article of technical interest and depth. A Feature Article in THE OEM CORNER is still eligible for the HP 32-E calculator prize.

IF YOU'RE PRESSED FOR TIME...

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

THE MECHANICS OF SUBMITTING AN ARTICLE

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

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

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

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

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

With best regards,

The Editor

USER'S QUEUE

LOCUS CHANGES AND ADDITIONS

This article updates the Data Systems LOCUS Program Catalog (22000-90099). The following changes have been made in existing LOCUS programs.

The "RTE 2/3 Activity Profile Generator for 21MX-M/E Computers" has been revised to be RTE-IV Operating System and HP 1000 F-Series Computer compatible. The program is now available on 800 bpi magnetic tape:

22682-10942	800 bpi magnetic tape	$80.00

The "21MX Instruction Set Simulator" program has been revised. It is now available on 800 bpi magnetic tape:

22682-10965	800 bpi magnetic tape	$70.00

The "SORT--RTE2/3 FMGR FILE TYPE 2 ASCII, INTEGER, and REAL SORT" program package has been revised. It is now available as follows:

22682-13367	Minicartridge	$70.00
22682-10967	800 bpi magnetic tape	$70.00

The "Tell and Tell All Inter-Terminal Message Sending" program has been revised. It is now available on cassette:

22682-13369	Minicartridge	$50.00

The RTE 3070A Terminal Utility Subroutines Package, part number 22682-XXX76, has been revised. It is now available only on magnetic tape:

22682-10976	800 bpi magnetic tape	$80.00
22682-11976	1600 bpi magnetic tape	$80.00

The "DBLST — RTE2/3 Image Data Base Information Lister" program has been revised. It is now available on cassette:

22682-13380	Minicartridge	$35.00

The "Tape Cartridge Save and Restore Routine for RTE2/3" has been revised. It is now available on Minicartridge.

22682-13387	Minicartridge	$40.00

The "RTE2/3 Multi-Terminal Program Development and Execution Interface with Automatic Spooling" package is available only on magnetic tape:

22682-10991	800 bpi magnetic tape	$50.00
22682-11991	1600 bpi magnetic tape	$50.00

The new contributed programs listed below are now available in LOCUS. Contact your local HP sales office to order Contributed Library programs.

22683-XXX15 LIBRN

LIBRN is an RTE utility program for separating and combining binary relocatable programs and subroutines. LIBRN also may be used to list the names and optionally the entry and external declarations of relocatable programs/subroutines.

LIBRN processes relocatable programs from any input/output device supporting relocatable read/write and/or RTE FMP type 5 disc files.

Input commands may be issued from a keyboard device, any ASCII input device or from a FMP type 4 file.

The user will find LIBRN extremely useful in updating library paper tapes, magnetic tapes and disc files.

LIBRN may also be used to retrieve subroutines from the system disc library.

22683-13315	Minicartridge	$70.00
22683-10915	mag tape	$70.00

22683-XXX16 CLIST DISC USAGE DISPLAY

This program displays the information found in the track assignment table and displays the contents of the file manager directories of all mounted cartridges.

As a result of summing the track assignment table entries by owner the list is generally short, usually about 10 to 15 lines.

The cartridge list section produces one line for each mounted cartridge. Each line contains all the information available by the use of the CL and DL commands. In addition, the amount of available disc space, the amount of disc space contained in purged files, and the number of directory entries in use are displayed.

CLIST will execute on any RTE-II, RTE-III, or RTE-IV SYSTEM with any combination of 7900, 7905, 7920, or 7925 discs.

22683-13316	Minicartridge	$35.00

22683-XXX17 SCAN

SCAN is a Fortran program that supplies a terminal hard-copy function for Hewlett-Packard's RTE systems using the 2645 and 2648 terminals. SCAN emulates a terminal hard-copy device by interrogating the terminals. To get a copy of all of the terminals' contents type: RU,SCAN. SCAN will automatically send the cursor to the home position and transfer the display information to the system printer (LU 6). For a partial scan, type: "RU, SCAN,,P" (move the cursor to the first line to be copied before typing "RETURN"). SCAN will copy from that line up to but not including the line containing RU,SCAN,,P. In either case, a form feed is provided after the copy as an added convenience.

22683-13317	Minicartridge	$35.00

22683-XXX18 EXC-FLOPPY HP INTERCHANGE FORMAT

This program will write a file tb a floppy in the HP interchange format. Data is read from a non-disc device (such as mag tape or 2645 tape cartridge). The data may be blocked or unblocked on tape.

The program will create a Volume Label and Directory on the Floppy, de-block the input data if required and write the records to the floppy in the HP interchange format.

This program does not read HP interchange format at this time. However, this feature will be added in the near future.

22683-10918	800 bpi mag tape	$40.00
22683-13318	Minicartridge	$35.00

22683-XXX19 LOG-RTE VIRTUAL TIMER SUBSYSTEM

The RTE Virtual Timer Subsystem implements a virtual timing scheme for RTE systems. A virtual timer is an elapsed timer that measures CPU time used by a particular program, independent of the CPU time used by other programs in a multiprogramming environment. It attempts to time the CPU usage of a program as if it were executing in a single-user non-multiprogrammed system.

The subsystem consists of a driver that maintains timers for up to 20 programs, and an operator interface used to monitor the operation and status of the subsystem. Any user program can make calls to the driver to determine the amount of CPU time used by the caller. In addition, information on system idle time is kept.

22683-10919	800 bpi mag tape	$50.00

22683-XXX20 DBCPY DATA-BASE COPY UTILITY

This program copies an IMAGE/1000 data base to an empty data base. DBCPY can be used for back-up, recovery, and restructuring without a mag tape. It can be useful even if a mag tape is present since it does not abort the copy on finding an error in the destination data base.

22683-13320	Minicartridge	$50.00

22683-XXX21 VASP-VIRTUAL ARRAY SUBROUTINE PACKAGE

This Fortran subroutine package enables a Fortran programmer to handle very large multidimensional real or integer arrays. The arrays are stored and recalled from disk by calls to the subroutines in such a way that any array element is accessible using subscript notation similar to that of a standard FORTRAN array. Up to 5 arrays are allowed, any of which can be real or integer. The integer arrays can have up to 4,194,176 entries, the real arrays up to 2,097,088 — depending upon available disc space. Up to 4 dimensions are allowed each array. The complete package uses 3808 words of storage. Several other routines are included which can be used to create and access arrays stored permanently on the disk and to inspect the array contents for troubleshooting purposes.

22683-13321	Minicartridge	$50.00

22683-XXX22 BIORM

Program BIORM plots one month bio-rhythm charts. The program asks whether a large or small bio-chart is desired. Small is 28 X 43 characters wide and large is 55 X 75 characters wide. The necessary dates are requested and the chart is printed.

22683-13322	Minicartridge	$40.00

22683-XXX23 DVC12-CENTRONICS COMPATIBLE PRINTER DRIVER

DVC12 is a standard printer driver and performs essentially the same functions as DVA12. An extension of the feature of printer controls by the first character of the user's buffer is included. Several parameters can be defined by changing the subchannel number of each device's EQT.

The driver supports all CENTRONICS compatible printers, and uses the 12566B interface, positive or ground true.

Parameters include:

1. No. of lines per page
2. No. of characters per line
3. Simulate display functions
4. Ignore control word Bit 8 (print first col)

22683-13323	Minicartridge	$35.00

USER'S QUEUE

LETTER TO THE EDITOR

Editor:

I have missed the Documentation Subsection of the Bulletins Section in the last several issues. I found this information quite useful in the past, both for determining the current version data of a manual and for finding the prices when I wanted to order manuals. Please resume publication of this feature as soon as you can.

The whole magazine has been useful to me, particularly the technical articles.

Keep up the good work.

Sincerely,

Richard B. Gilbert

Dear Sir,

We have found that it is difficult to keep an up-to-date list of our documentation in the COMMUNICATOR/1000. One cause for this is that manual updates do not correspond to the publication cycle of the COMMUNICATOR/1000. However, the Software Update Notice (SUN) does contain this information, and the SUN's publication cycle does correspond to that of the manual updates. It is the best place to find accurate and timely manual update information. All customers on Software Notification Service, Software Subscription Service or Comprehensive Software Support receive the SUN quarterly as well as the COMMUNICATOR/1000. We are glad you find the COMMUNICATOR/1000 useful. We feel that the technical articles are what makes the COMMUNICATOR/1000 a viable and valuable publication. Perhaps you or someone in your group would like to contribute an article?

With best regards,

The Editor

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

Dear Samantha,

Is there any way to patch the default line count in the Assembler and in the FORTRAN Compiler? We use fifty-line printer paper and would like to be able to do this.

While we're on the subject of default parameters, is there a specific reason for making the default priority of programs 99 rather than another value (i.e. 10000)? How do I change the priority of a program?

Sincerely,

F. Stephen Gauss
Naval Observatory

Dear Stephen,

The default line count of the Assembler and the FORTRAN Compiler can be changed in a way that is nearly transparent to the terminal user. This method utilizes Transfer Files and Global Parameters. An example for the FORTRAN Compiler is shown below.

Utilizing EDITR, define a Transfer File, FTN, as shown:

```
:RU,FTN4,1G,2G,3G,50,5G
```

The fourth parameter in the FTN Transfer File defines the line count on the list output to be 50 lines. The Global Parameters 1G through 5G allow parameters such as the source file name to be passed to the Compiler.

When you desire to compile a FORTRAN program whose source is contained in &MYFIL, type in

```
RU,FTN,&MYFIL,6,-
```

This will cause the contents of Transfer File FTN to be executed. The source contained in &MYFIL will be compiled, the compiler output will be listed in 50 line pages on LU 6 (instead of 56 line pages), and the relocatable code will stored in file

%MYFIL. &MYFIL has been passed through 1G, 6 through 2G, etc.. Global Parameters allow the first six characters in a string between commas to be passed. If you desire to pass more than six characters, such as with

```
        &MYFIL::700
```

in order to specify that &MYFIL resides on disc cartridge 700, then you should write a short program to pick up more than six characters and schedule FTN4 (passing the desired line count).

Concerning your other question, there is probably no scientific reason for setting the default priority to 99. Priority can be changed through use of the RTE "**PR**" command.

Thank you for your interest.

Sincerely,

Samantha

STOP USING TRICKS TO RESOLVE INDIRECT EXTERNAL DEF'S

Glenn Talbott/HP Data Systems Division

Many of Assembly Language programmers (the author included) have gotten into the bad habit of using tricks to get the Loader to resolve the direct addresses of DEF's to external references, which are usually indirect; and some of us have gotten burned. A common trick has been to reference the external with a zero offset, and the Loader was nice enough to give us the direct address instead of the indirect address that is returned when no offset is specified. Just try relocating code that has used the zero offset trick using the Generator instead of the Loader. The Generator will create an indirect link when the offset on an external DEF is specified as zero, while the program code is expecting a direct address. This code may then be executed with unpredictable results.

Unbeknownst to many of us, Hewlett-Packard does not guarantee a direct address from a DEF, or a specific level of indirects. This applies to DEF's to program relocatable locations as well as the above mentioned external DEF's. The levels of indirects returned today may change with tomorrow's operating systems.

Whenever a direct address is required the RTE Library Indirect Address Subroutine, **.DRCT**, should be used. The calling sequence is as follows:

```
EXT .DRCT
JSB .DRCT
DEF ADDR
-return-
```

The routine returns with the A-Register set to the direct address of ADDR, the B-Register is unaltered, and the E-Register is lost.

The little bit of overhead incurred using **.DRCT** is a small price to pay when you consider the man-hours involved tracing down the kind of problems produced when tricks like zero offset backfire; not to mention the benefits of keeping your software compatible with the Loader, Generator, and future HP software.

RELOCATABLE DRIVERS FOR RTE-II

John Blommers
Defense Research Establishment Pacific
Bldg 199, CFB Esquimalt
Victoria, British Columbia
Canada V0S1B0

Introduction

The RTE-II operating system drivers, along with the rest of the operating system, are permanently core resident. There is no facility to permit the addition of drivers once the operating system has been built by the system generator program. This makes the debugging of newly written drivers very inconvenient, since a new sysgen must be performed to test them.

It would be desirable to load a user program along with the new driver into a memory partition. Because the driver disappears when the user program execution is terminated, memory which otherwise would be permanently occupied by the driver now available for general use. To direct RTE to initiate I/O correctly, a selected EQT entry must be rebuilt to reflect the characteristics and requirements of the new device.

Rebuilding an EQT Entry

Since all I/O devices must have an associated EQT entry, some existing EQT entry must be sacrificed and rebuilt for the new device. Unless the user is prepared to first save, and subsequently restore the selected EQT entry, the EQT should be a spare, or an infrequently used one. (Rebooting the system will, of course, restore the EQT to the state specified in the original system generation.)

Before doing any I/O to the new device (be it via a FORTRAN READ/WRITE, a call to REIO, or a call to EXEC), the user program must rebuild the selected EQT entry. This is a two-step process. The first step is to set the second and third words of the EQT entry to the driver initiator and continuator addresses respectively. The second step is to rewrite words 4,5 and 14.

Setting Words 2 and 3 of the EQT Entry

The RTE operating system requires that the driver initiator and continuator addresses be present in words 2 and 3 respectively of the EQT entry. In this way, RTE can pass control to the appropriate driver section when an interrupt from the device controller (or its associated DMA channel) occurs. Since the driver may be located anywhere within the user partition, words 2 and 3 must be adjusted each time the user program is executed.

A simple FORTRAN callable assembly language subroutine must be called first. It contains two external references — one to I.XX (the initiator address), and one to C.XX (the continuator address). Clearly the driver dvrxx must declare I.XX and C.XX as entry points. A suitable subroutine is shown in Figure 1.

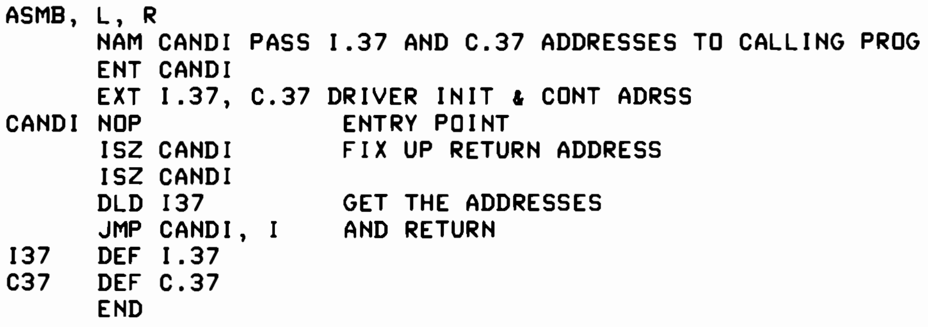
CANDI simply returns the two addresses (or two indirect references to two basepage links) in the A and B registers. Thus CANDI must be used as a REAL FUNCTION in the FORTRAN user program, as shown in Figure 2.

The function IGET returns the contents of the address specified in its argument. The function IPUT stores the second argument into the address specified by the first. (IPUT is a priviledged subprogram calling $LIBR and $LIBX, thereby enabling it to write below the memory protect fence).

Setting Up Word 4 of the EQT Entry

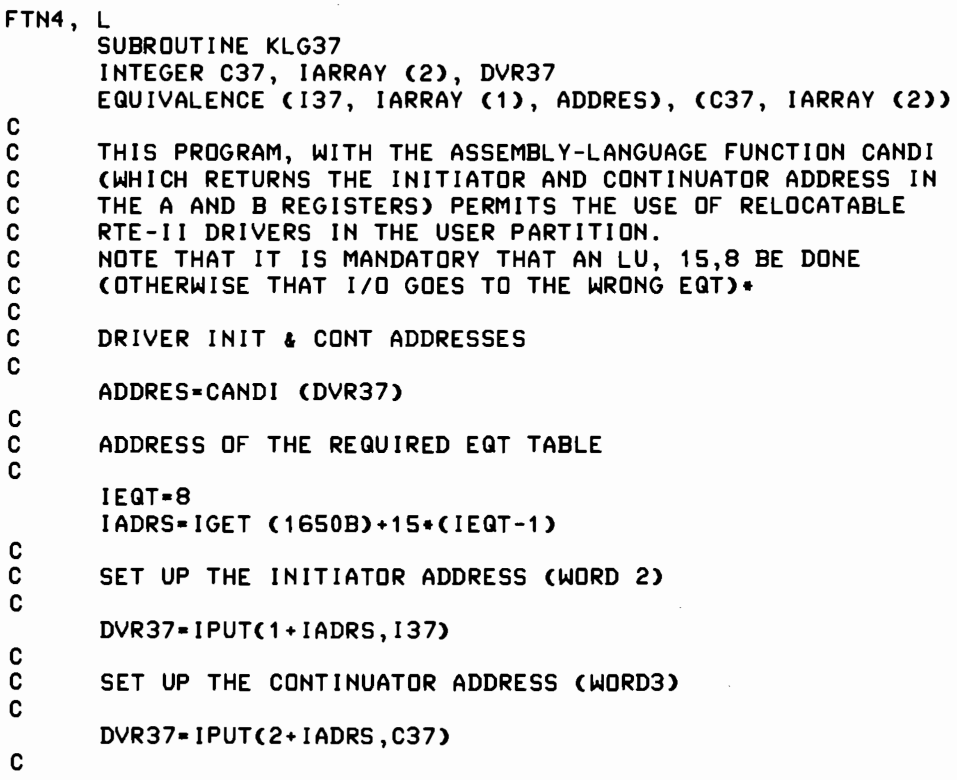
The definition of word 4 is described in Figure 3.

Figure 1



```
ASMB, L, R
      NAM CANDI PASS I.37 AND C.37 ADDRESSES TO CALLING PROG
      ENT CANDI
      EXT I.37, C.37 DRIVER INIT & CONT ADRSS
CANDI NOP             ENTRY POINT
      ISZ CANDI       FIX UP RETURN ADDRESS
      ISZ CANDI
      DLD I37         GET THE ADDRESSES
      JMP CANDI, I    AND RETURN
I37   DEF I.37
C37   DEF C.37
      END
```

Figure 2



```
FTN4, L
      SUBROUTINE KLG37
      INTEGER C37, IARRAY (2), DVR37
      EQUIVALENCE (I37, IARRAY (1), ADDRES), (C37, IARRAY (2))
C
C     THIS PROGRAM, WITH THE ASSEMBLY-LANGUAGE FUNCTION CANDI
C     (WHICH RETURNS THE INITIATOR AND CONTINUATOR ADDRESS IN
C     THE A AND B REGISTERS) PERMITS THE USE OF RELOCATABLE
C     RTE-II DRIVERS IN THE USER PARTITION.
C     NOTE THAT IT IS MANDATORY THAT AN LU, 15,8 BE DONE
C     (OTHERWISE THAT I/O GOES TO THE WRONG EQT)*
C
C     DRIVER INIT & CONT ADDRESSES
C
      ADDRES=CANDI (DVR37)
C
C     ADDRESS OF THE REQUIRED EQT TABLE
C
      IEQT=8
      IADRS=IGET (1650B)+15*(IEQT-1)
C
C     SET UP THE INITIATOR ADDRESS (WORD 2)
C
      DVR37=IPUT(1+IADRS,I37)
C
C     SET UP THE CONTINUATOR ADDRESS (WORD3)
C
      DVR37=IPUT(2+IADRS,C37)
C
```

*Logical Unit 15, EQT 8 are used in this example.

Figure 3

EQT Word 4

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D	B	P	S	T	UNIT NO.					CHANNEL NO.					

D=1 if DMA is required.
B=1 if automatic output buffering is used.
P=1 if the driver is to process power fail.
S=1 if the driver is to process time-outs.
T=1 if the device has timed out. (The system sets
this to zero before each I/O request.)

UNIT NO. is the last sub-channel addressed.
CHANNEL NO. is the I/O select code for the device
(lower number if a multiboard interface).

If the driver always requires a DMA channel, then bit 15 should be set. In this case, RTE will assign a DMA channel to the driver (via basepage location CHAN) before passing control to it. If no DMA channel is available, a request for one is queued up and the calling program is suspended. If the driver does not require DMA, then bit 15 is clear. The requirement for DMA may exist only when large data buffers are to be moved, since short buffers may be moved efficiently on an interrupt basis. When the driver decides that it needs a DMA channel, it may exit from its initiator section with the A-register = 5. RTE will immediately attempt to assign a DMA channel for the driver and return control to it if one is available. Otherwise the DMA request is queued up as before.

Bit 14 must always be zero when relocatable drivers are used. This prevents automatic output buffering, a feature which permits a program to be swapped with a higher priority program. If the user program is swapped, the driver disappears, and while the RTE attempts to perform the I/O from SAM, unpredictable events will occur when control is passed to the address where the driver was previously located. By avoiding automatic output buffering, the user program cannot be swapped during I/O, since this causes the buffer to disappear while it is still required by the driver.

It is interesting to observe that, since both the buffer and the driver exist within the same user program, it is logically possible to swap the program during non-DMA I/O. Whenever there is an interrupt from the device, and the program is swapped out, the program must be restored to service the interrupt. Whether this process is fast enough or not depends on the device in question. (RTE-II does not operate this way.)

Bits 13 and 12 are individually set to one by the driver to inform the system that either power-fail and/or time-out are to be processed by the driver. Thus bits 13 and 12 should both be zero at this point.

Bit 11 should be clear. It is set to one by RTE when the device has timed out.

The UNIT NO. may be set to 0. UNIT NO. is set up by the system according to the subchannel number associated with the EQT number associated with the logical unit (LU) to which the I/O is directed.

The CHANNEL is the octal select code of the device, and must be set up correctly.

The following is a FORTRAN statement (part of the previous subroutine) for setting up word 4:

```
C  SET UP THE SELECT CODE (WORD 4)
C
   CALL IPUT(3+IADRS, 25B)
C
```

Setting Up Word 5 of the EQT Entry

The definition of word 5 is described in Figure 4.

Figure 4

EQT Word 5

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AV		EQ TYPE CODE						STATUS							

AV is the Availability indicator

0 = Available for use.
1 = Disabled (down).
2 = Busy (currently in operation).
3 = Waiting for an available DMA channel.

STATUS is the actual physical status or simulated status at the end of each operation.

EQ TYPE CODE is the type of device. When this octal number is linked with "DVx" it identifies the device's software driver routine as follows:

00 to 07	=	Paper tape devices (or system control devices).
00	=	Teleprinter (or system keyboard control device).
01	=	Photoreader.
02	=	Paper tape punch.
05 sub 0	=	Console (or system keyboard device).
05 sub 1 05 sub 2	=	Mini-cartridge devices.
10 to 17	=	Unit record devices.
10	=	Plotter.
11	=	Card reader.
12	=	Line printer.
15	=	Mark sense card reader.
20 to 37	=	Magnetic tape/mass storage devices.
30	=	Fixed-head disc, or drum.
31	=	7900 moving head disc.
32	=	7905, 7920 moving head disc.
40 to 77	=	Instruments.
CONWD	=	User control word supplied in the I/O EXEC call.

The AV and status fields may be set to 0 since they are used by the RTE and the driver to report status conditions.

The EQT TYPE CODE should be set to XX (as in DVRXX).

To set up words, the following FORTRAN sequence may be used (for DVR37).

```
C   SET UP THE EQT TYPE (WORD 5)
C
    CALL IPUT(4+IADRS, 017400B)
```

Setting Up Word 14 of the EQT Entry

This word contains the time-out value, in tens of milliseconds. When the driver exits from the initiator section, the A-register is set to one of the following:

0 – operation successfully initialized;
1 – read or write request is illegal for device;
2 – control request illegal or undefined;
3 – equipment malfunction or not ready;
4 – immediate completion of operation;
5 – driver requires DMA but bit 15 in EQT not set.

If A=0, word 15 of the EQT is cleared by RTE, and is incremented on every TBG interrupt. Should word 15 ever equal the time-out value, the device would have timed out, since it had not interrupted in the expected time. Whenever the device interrupts, control is passed to the driver continuator section, and upon exit from it, word 15 is cleared again.

The time-out is a negative number and may be set up as follows (time-out 1 second).

```
C   SET THE TIME-OUT VALUE (WORD 14)
C
    CALL IPUT(13+IADRS,-100)
```

Setting Up the Trapcell

When the user has determined which select code the new device will possess, he must install a JSB $CIC instruction in the trapcell. The location of this entry point varies from system to system, and may be found by the following assembler language subprogram in Figure 5.

Figure 5

```
ASMB, R, L
      NAM GETIT
      ENT GETIT
      EXT $CIC,.ENTR
GETIT NOP
      JSB .ENTR
      DEF GETIT
      LDA JSB
      JMP GETIT, I
JSB   JSB $CIC
      END
```

The following FORTRAN code will set up the trapcell (25B in this example):

```
    INTEGER GETIT
    CALL IPUT (25B, GETIT (DUMMY))
```

If the EQT entry being sacrificed isn't a spare, but isn't used during the time the user program is running, then the entire EQT entry plus the trapcell must be saved prior to the rebuilding process, and restored just before the user program terminates.

Setting Up the Device Reference Table

Since all I/O is directed to logical units, the user must assign some available LU to the sacrificed EQT entry. The simplest way to do this is with an FMGR directive (which may be stored in a transfer file) as follows:

```
        :SYLU,15,8
```

Summary

By loading a driver with the user program and sacrificing an EQT entry, new drivers may be debugged easily. The user program rebuilds the EQT entry and the trapcell, optionally restoring them later. Automatic output buffering of the EQT entry must be disabled.

Foreground Disc Resident Drivers (A Concept)

If a foreground partition of sufficient size is present in the system, then a driver may be loaded along with a small EQT building routine. The foreground partition must not be swappable (:SYSW,2 must be done as defined by the basepage address SWAP) or else the driver will disappear. Further, no other foreground disc resident programs must enter the partition after the driver is installed there and after it has terminated (core lock not applicable for dormant programs).

The user program wishing to perform I/O to the new device merely schedules the I/O to the device, since the driver cannot disappear.

Note that the regular disc resident foreground programs won't run with the driver in place. If required, these must be executed in the background partition.

WHAT IS DISC TRACK SPARING?

Bob Sauers/HP Rockville

Disc track sparing is an often misused and misunderstood term, but is important to any RTE system manager. This article assumes that the reader is familiar with RTE system generation and system backup procedures. The term disc track sparing, or simply sparing, is a term which is referred to during normal system operation, during system backup, and when using the on-line system switch program "SWTCH". Hopefully, this article will clear up some misconceptions regarding sparing.

A More Precise Definition

The most widespread usage of the term sparing refers to the process of detecting a bad track during a read or write operation on the disc, flagging the track defective, assigning a spare track, and finally, utilizing that track. Although this definition is prevalent, it is not strictly accurate.

More precisely and simply, sparing is the act of automatically seeking to a spare track upon detection of a defective track which cross references itself to the spare track. This process may be initiated either by the software driver or the controller. Whether the software driver or controller performs the sparing function depends upon the nature of the disc drive and the "intelligence" of the controller.

To coin a new term, perhaps, the process of detecting a defective track, flagging it defective, and assigning and utilizing a spare track in its place can be called **activating a spare.**

The term flagging refers to initializing or formatting the disc track and is not the same as the File Manager initialize command (`IN,...`).

Referring to the precise definition of sparing, it can be seen that sparing can occur during normal RTE system operation, if it is enabled. Recall that disc track maps are defined during system generation. Spare tracks are assigned only for the 7905/06/20 family of compatible disc drives. Therefore, the process of sparing is performed only on these disc drives. Although sparing does not occur on 7900/01 disc drives, it is useful to discuss the process required for sparing on these drives to fully appreciate the capabilities of the more sophisticated disc drives and controllers.

The Layout of the Disc

The use of the word "sector" in this article refers to a physical sector on the disc of 128 data words. RTE refers to a physical sector as a block. In RTE, a sector is 64 data words and should be thought of as a logical sector.

In order to fully understand sparing and how sparing is enabled, the structure of a disc track will be discussed. Although a disc track is circular, and is separated radially into a number of sectors, it is easier to picture the track as a straight line (see Figure 1). Each physical sector in the track is actually separated into three sections: the preamble, data, and postamble (Figure 2). The preamble is used for synchronization and addressing by the disc drive. The preamble contains information necessary to the process of sparing — namely, the track status flag bit(s) and the head/sector and cylinder address fields (Figure 3). The data field contains the actual data transferred to or from the computer. The postamble contains sector check words which are used for error detection and error correction on some disc drives.

Sparing on 7900/01 Disc Drives

Sparing on the 7900/01 disc drives must be a manual operation, performed entirely by the software driver. Each platter of the 7900 disc drive or the single platter of the 7901 disc drive is a subchannel. Each subchannel contains 48 sectors, 24 sectors on each surface. The preamble of a sector on the 7900/01 disc drives (see Figure 4), contains one bit which serves as a track status bit, indicating that a given track is defective or protected. Also, only one bit in the disc status word is available to indicate

Figure 1 — Generalized Disc Track Format

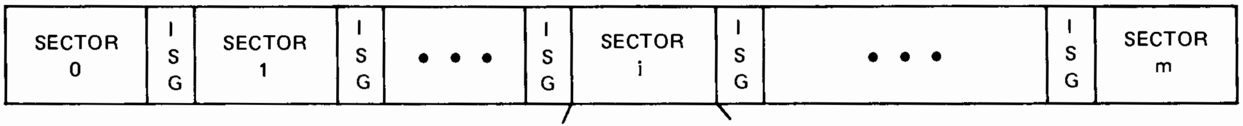


Figure 2 — Disc Sector

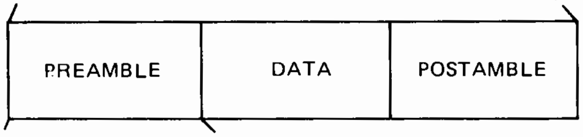


Figure 3 — Sector Preamble

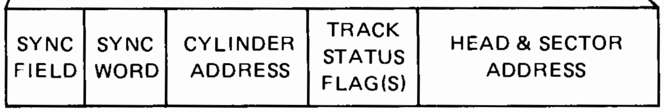
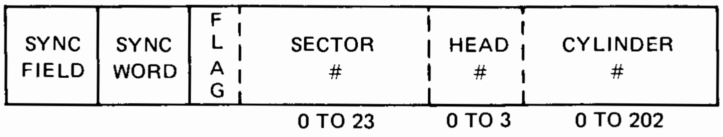


Figure 4 — 7900/01 Disc Drive Preamble



FLAG=1: PROTECTED OR DEFECTIVE TRACK

track status (Figure 5). That bit is the flagged track bit. Whenever a disc drive seeks to a specified track and sector, the preamble is read, and the head, sector, and cylinder addresses are sent to the controller. The controller compares the addresses sent from the disc drive and the addresses sent from the computer. If the two sets of addresses compare and the flag bit is set, then the track is protected. If the two sets of addresses do not compare, then the controller sets the address error bit in the disc status word, and the track is considered defective. Therefore, a track which is flagged defective must contain an incorrect address.

Figure 5 — 7900/01 Disc Controller Status Word

X	FIRST STATUS	OVER-RUN	X	UNSAFE	DATA PROTECT	X	SEEK CHECK	X	NOT READY	END OF CYL	ADDRESS ERROR	FLAGGED CYL	DRIVE BUSY	DATA ERROR	ANY ERROR

If, during a read or write operation, the 7900/01 disc controller detects a defective track, it informs the software driver by setting the flagged track and address error bits in the disc status word (which the software driver should request). Assuming that track sparing is utilized, the address stored in the address field of a defective track would be that of the spare track. The software driver would then issue a read address request to the controller to retrieve the address of the spare track. A new seek request to the address of the spare track would be issued by the software driver to then access the spare track. One might ask that if the track is defective, how can the spare track address be read from the sector preamble? Typically, a defective track is defective in only one or several sectors and not the entire track. Therefore, the software driver would continue to attempt to read the disc address field from successive sectors until it found one it could successfully read. The 7900/01 disc drives have twenty-four sectors in a single track, and therefore, the software driver might conceivably have to read the address fields of all twenty-four sectors. This process, which is sparing, is very complicated on the 7900/01 disc drives and is not really so automatic.

A software driver must be written to perform all of the functions indicated above. In RTE, the 7900/01 disc driver (DVR31) does not perform these functions and therefore, sparing is not enabled on these disc drives in RTE.

On a 7900/01 disc drive, if RTE encounters an unflagged defective track (detected by a transfer error), it will only report the error with a `TR nnn` message and will not activate a spare or even merely flag the track defective. If the defective track occurs in the file management area of the disc, FMGR can be instructed not to use the defective track by using the `IN,...` command. The logical track number `nnn` from the `TR nnn` message should be specified as one of possibly up to five "bad track" parameters of the `IN,...` command. If RTE encounters a flagged defective track, it also generates a `TR nnn` error message in a 7900/01 disc-based system. Note that a `TR nnn` error message can be generated when attempting to write on a track which is flagged protected with format disabled (protect/override switch in protect position). An example of that is using the loader (LOADR) to make a permanent addition to the system (i.e., `RUN,LOADR,,%name,,PE`).

Sparing for the 7905/06/20 Family

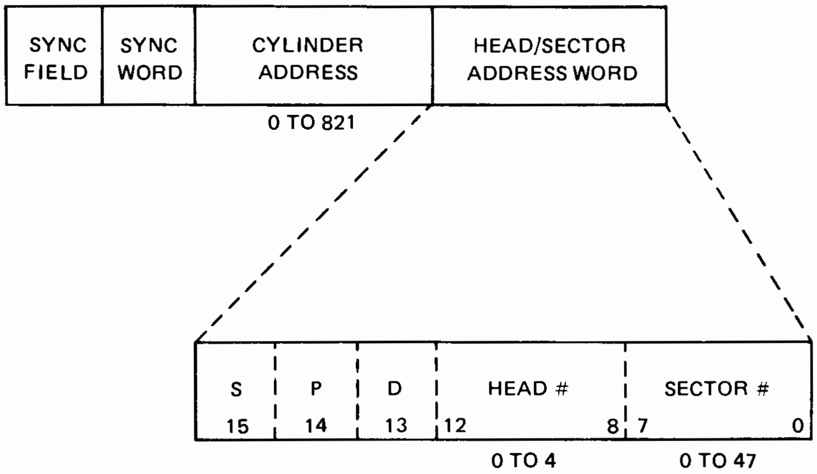
For the 7905/06/20 family of compatible disc drives, the process called sparing is much simpler because the disc controller is "intelligent." The sparing process, if enabled, is also much more rapid because it is performed entirely by the disc controller without intervention of the computer and software driver. Recall that the track map tables for these disc drives, built during system generation, contain information pertaining to spare tracks. Although RTE itself does not utilize the spare track information during normal system operation, the information is necessary in order for track sparing to be enabled.

The disc sector preamble of a 7905/06/20 disc drive is more complicated than the preamble on the 7900/01 disc drives. The only difference important to this discussion, is that the 7905/06/20 disc drive preamble contains three bits, rather than one, for track status. These are: protected (P); defective (D); and spare (S). Figure 6 shows the general format of the track preamble for a 7905/06/20 disc drive.

The sector preamble of track 147, head 2, and sector 35, assuming that it were unflagged, would look like Figure 7. If that same track were flagged defective and cross referenced to a spare one, for example, track 257, then both tracks would look like Figure 8. Note that an entire track must be flagged, even though only one sector may be defective. Therefore, the cross reference between defective and spare is done on a track basis.

The term "if enabled" is used when referring to the sparing process on 7905/06/20 disc drives. The disc contoller (13037A/B/C) must be instructed to enable automatic sparing. Upon RTE system bootup, the disc controller receives a "cold load" command from the program stored in the disc loader ROM, which causes the disc controller to assume a default state. An important part of the default state is that automatic sparing is enabled.

Figure 6 — 7905/06/20 Disc Drive Preamble



S=1 FOR SPARE TRACK IN ACTIVE USE
P=1 FOR PROTECTED TRACK
D=1 FOR DEFECTIVE TRACK

Figure 7 — Sample Preamble for 7905/06/20 Disc Drive Cylinder 147, Sector 35, Head 2

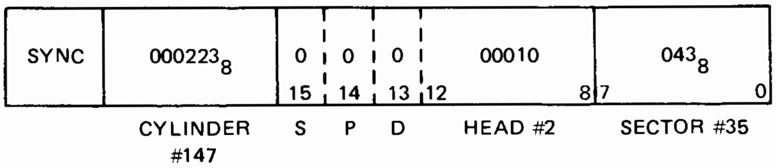
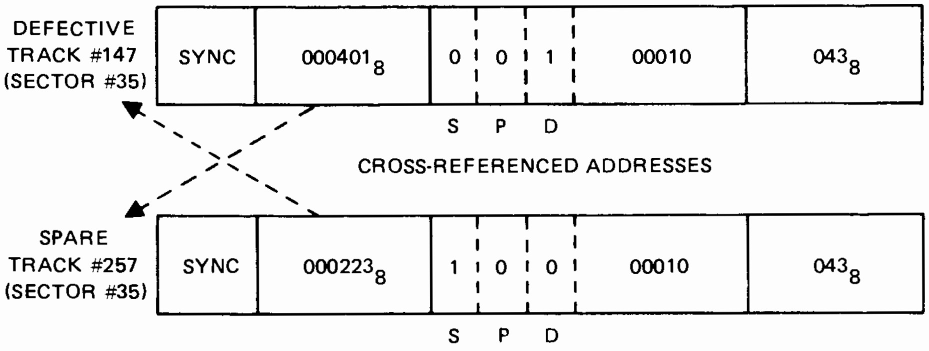


Figure 8 — Sample Preambles of Cross-Referenced Defective & Spare Tracks


DEFECTIVE TRACK #147 (SECTOR #35)
SYNC
000401_8
0
0
1
00010
043_8
S
P
D
CROSS-REFERENCED ADDRESSES
SPARE TRACK #257 (SECTOR #35)
SYNC
000223_8
1
0
0
00010
043_8
S
P
D

When the disc controller receives a seek command from the software driver preceeding a read or write operation, it reads the preamble of the sector to determine the status of the track. If the track is flagged defective (the D-bit is set to one), and automatic sparing is enabled (as in RTE), the controller automatically utilizes the head/sector and cylinder address field to determine the address of the spare track. The controller then issues a new seek command to the disc drive to access the spare track. The address field in the spare track is, of course, cross-referenced back to the defective track, and the S-bit is set in the track status field to indicate that a spare track is in active use.

It can be seen that the disc controller for a 7905/06/20 disc drive assumes all of the work when automatic sparing is enabled. RTE and the software driver, DVR32, are never informed that sparing has occurred.

If RTE encounters an unflagged defective track on a 7905/06/20 disc drive by detecting a data transfer error, RTE will report the transfer error with a **TR nnn** message just as with a 7900/01 disc drive. However, if RTE encounters a defective track on a 7905/06/20 disc drive which was flagged defective and a spare track has been activated, then sparing will occur.

In all of the preceeding discussion, no mention has been made as to how the spare disc tracks are activated. Although automatic sparing is enabled on the 7905/06/20 disc drive controllers during normal RTE system operation, the process of flagging tracks (either protected, defective, or spare) does not occur in normal RTE system operation.

The reason that the process of activating spare tracks does not occur during normal RTE system operation is that the disc drivers — DVR31 for the 7900/7901 drives, and DVR32 for the 7905/06/20 disc drives — do not contain the necessary logic required to activate spare tracks. It would likely be extremely difficult to activate spare tracks in a multiprogramming environment since manual intervention would be required to switch to override on 7900/01 drives or format enable on the 7905/06/20 drives. Whatever the reasons, spare track activation does not occur on-line in RTE.

Activating Spare Tracks

There are three supported programs in RTE which activate spare tracks. Remember, that activating spare tracks means: flagging a track defective, assigning a spare track, and then utilizing the spare track. These programs are:

1. The disc diagnostic (manually, through operator dialog)
2. The off-line system backup utility (!DSKUP)
3. The on-line system switch utility (SWTCH).

The disc diagnostic is rather cumbersome for uninitiated users to utilize for activating spare tracks. The diagnostic configurator must be loaded and configured. Then the disc diagnostic must be loaded and executed with the "op design" bit set in the switch register. Then, a small "op design" program must be written to perform the spare track activation.

The off-line system backup utility, !DSKUP, will activate spare tracks only if the "save" was to magnetic tape and was either an on-line "LU" save or an off-line "UNIT" save with sparing. In both cases, the track map table is saved along with the data so that the "RESTORE" can assign spare tracks where necessary. In the case of the off-line unit save with sparing, it is assumed that the operator correctly typed in the track map table during the save operation.

Since SWTCH is an on-line RTE system switch utility program, one might assume that SWTCH performs spare track activation where necessary by acting through DVR32. Not so! The system switch utility incorporates a self-contained disc driver which has more capabilities than DVR32. Also, RTE is actually disabled during the time of the system switch. SWTCH automatically performs track flagging and spare track assignment on the system subchannel (Logical Unit TWO). Track flagging and spare track assignment may be performed on other subchannels by answering "yes" to the question initialize subchannel "n" where "n" is the number of a subchannel other than the system subchannel. Note well that all data previously stored on subchannels which are initialized will be lost.

The Important Points

The important points in this article can be summarized as below.

- Sparing occurs only on 7905/06/20 Disc Drives and not on 7900/01 drives.
- Sparing occurs during normal RTE system operation.
- Sparing is not track flagging and spare track assignment.
- Track flagging and spare track assignment is performed only by the disc diagnostic, the off-line system backup utility, and the on-line system switch utility.

Hopefully, this article has been useful in clearing up misconceptions pertaining to sparing and has given the reader a better understanding of the sparing and spare track activation processes in RTE.

REFERENCES:

1. Installation and Service Manual for Model 13037A/B Disc Controller, June 1978, part number 13037-90006.
2. Operating and Service Manual for 13210A Disc Drive Interface Kit, May 1978, part number 13210-90003.
3. Operating and Service Manual for 7900A Disc Drive, February 1977, part number 07900-90002.
4. Operating and Service Manual for 7920A Disc Drive, August 1978, part number 07920-90001.

CREATING AND CLEARING EXTENTS

Alan K. Housley/HP Data Systems Division

Extent Creation

Anyone who has used the file manager DL command any number of times has surely noticed some of the file names are followed by a three-digit number to the right of the file's block size. These numbers signify that the file entry is an extent of a main file of the same name.

```
NAME    TYPE  BLKS/LU OPEN TO
          .
          .
          .
EXAMPL 00004  00048
EXAMPL 00004  00048  +001  <---- File extension #1
OUTPT  00004  00020              for file EXAMPL.
NEEDS  00004  00010
OUTPT  00004  00020  +001
OUTPT  00004  00020  +002
SYSMGR 00004  00001
CODES  00004  00005
FILES  00003  00017
SYSMGR 00004  00001  +001
          .
          .
          .
```

An extent (short for file extension) is automatically created by FMP with the same name and size as the original file (files type-3 and above) when a write request points to a location beyond the range of the currently defined file. Each extent is identified by an extent number which is stored in the upper byte of word five of the corresponding entry in the file directory. Whenever another extent is created by FMP, the extent number is incremented and stored in the file directory entry as described above.

EXAMPLE: Shown in the figure on the next page, is the file directory entry for the file **EXAMPL:11156:20:3:1,** and its first file extension.

```
Word #
       +-----------------------+     +-----------------------+
   0   ! File Name Word 1 = EX !     ! File Name Word 1 = EX !
       +-----------------------+     +-----------------------+
   1   ! File Name Word 2 = AM !     ! File Name Word 2 = AM !
       +-----------------------+     +-----------------------+
   2   ! File Name Word 3 = PL !     ! File Name Word 3 = PL !
       +-----------------------+     +-----------------------+
   3   !     File Type = 3     !     !     File Type = 3     !
       +-----------------------+     +-----------------------+
   4   !  Starting Track = 173 !     !  Starting Track = 174 !
       +-----------------------+     +-----------------------+
   5   !   Extent  !  Starting !     !   Extent  !  Starting !
       !  # = 0    !Sector = 20!     !  # = 1    !Sector = 22!
       +-----------------------+     +-----------------------+
   6   !    # of Sectors = 2   !     !    # of Sectors = 2   !
       +-----------------------+     +-----------------------+
   7   !   Record Length = 0   !     !   Record Length = 0   !
       +-----------------------+     +-----------------------+
   8   ! Security Code = 11156 !     ! Security Code = 11156 !
       +-----------------------+     +-----------------------+
   9   !           .           !     !           .           !
   .   !           .           !     !           .           !
   .  ~~~          .          ~~~   ~~~          .          ~~~
  15   !           .           !     !           .           !
       +-----------------------+     +-----------------------+
```

File Directory Entries For File EXAMPL

FMP will repeat the process of creating extents as many times as necessary as a file increases in size. Only a full disc LU (FMGR -006 error), a full file directory (FMGR -014 error), or an extent number exceeding 255 (FMGR -006) will abort the process.

Everytime a file or part of a file that contains extents is accessed, the file directory entry for each extent must be read in order to find the location of the file extension on the disc. This of course slows down the file access process. Obviously, to speed up the process, all file extensions might be combined into one file and therefore only one entry from the file directory would have to be read to locate the entire file.

Extent Cleanup

To combine file extensions into one continuous file is a simple process of storing the file into a temporary file, purging the original file, and renaming the temporary file to the original file name. As yet, there is not a supported routine that clears up file extensions in this manner, so many people have written transfer files for their personal use that clear extents on their disc LU. One version of such a routine is:

```
:ST,File Name Containing Extents:SC:CR#,Temporary File Name:SC:CR#::-1,SA
:PU,File Name Containing Extents:SC:CR#
:RN,Temporary File Name:SC:CR#,Original File Name
```

Put in the form of a transfer file using globals, this routine will process one file at a time. However, when several files need to be cleared of extents, the user must obtain all the file names by using the DL command and enter each file name in the transfer file one at a time. I personally find this process tedious, and consequently wrote a program to find all files with extents for a particular disc LU and clear them. I can simply run the program from time to time to keep my disc LU "cleaned up", and I don't need to know which files have gained extents since the last "clean up".

The CLEXT Extent Clearing Utility Program

Shown at the end of this article is the routine that will clear a disc LU of extents. Before running the program two other tasks must be completed:

1) As shown in line 156 of the program, FMG06 is scheduled to perform the ST command as discussed in the transfer file above. Therefore, a copy of FMG06 must be created as follows:

```
:RN,FMGR,FMG06
:RP,FMG06
:RN,FMG06,FMGR
```

   It would be useful to place this sequence in the WELCOM file, so a copy will always be present. I chose FMG06 because a copy of FMGR must correspond to an I/O device. FMG06 is the copy used by the printer and is probably never used by anyone. If for some reason anyone uses FMG06, create a copy that corresponds to some other "output only device", such as a magnetic tape unit, and change the 0 and the 6 in line 32 of the program to the new value.

2) Edit a file to contain only one line of text. This line is simply:

```
:EX
```

   Name the file `FILENM::-2`.

   The purpose of this file is to cause an exit from FMG06 after the ST has been completed as shown in line 156. For more information about the scheduling of FMGR programatically, see the RTE-IV Batch Spool Monitor Manual.

Final Notes

When running the program, only one user response is required: the disc LU to be cleared when prompted by CLEXT. If 0 is entered, CLEXT will abort. If LU 2 or if LU 3 is entered, CLEXT will prompt for another LU. If for some reason the store is unsuccessful (line 156), the program will abort. At this time the user should check to see if the disc LU is full, if the file directory tracks are full, or if the temporary file already exists. After fixing any of these problems, CLEXT may be run again to "clean up" the rest of the extents.

```
0001  FTN4,L
0002        PROGRAM CLEXT(),AKH'S    CLEAR EXTENTS
0003  C
0004  C*********************************************************************
0005  C                                                                   *
0006  C  NAME:     CLEXT (CLEAR EXTENTS)                                  *
0007  C                                                                   *
0008  C  DATE:     01/11/79                                               *
0009  C                                                                   *
0010  C  VERSION:  AKH.01                                                 *
0011  C                                                                   *
0013  C            DSD TECHNICAL MARKETING                                *
0014  C                                                                   *
0015  C  PURPOSE:  ROUTINE TO CLEAR EXTENTS FROM AN ENTIRE DISC LU.       *
0016  C                                                                   *
0017  C  INVOKE:   RU,EXTPU                                               *
0018  C                                                                   *
0019  C*********************************************************************
```

```
C
C
      INTEGER AB, BUFFST(30), COUNTR, CRNUM, DIRBUF(16), DLU
      INTEGER DRTRK1, FILDIR(128), FMGR(3), IARRAY(3), IDCBP(144)
      INTEGER IDCBS(144), IDCBT(144), NAMEP(3), NAMET(3), SECTRK
      INTEGER STATUS(125), TEST, TKCNTR, W, X, Y, Z
C
      DATA BUFFST / 2H:S, 2HT,, 2H00, 2H00, 2H00, 2H: , 2H00, 2H00,
     +2H00, 2H: , 2H00, 2H00, 2H00, 2H, ,
     +2H##, 2H.T, 2HMP, 2H: , 2H00, 2H00, 2H00, 2H: ,
     +2H00, 2H00, 2H00, 2H::, 2H-1, 2H, , 2HSA/
C
      DATA FMGR / 2HFM, 2HG0, 2H6 /
      DATA NAMET / 2H##, 2H.T, 2HMP/
C
C   OBTAIN THE INTERACTIVE LU AND THE DISC LU NUMBER.
C
      LLU=LOGLU(I)
 10   WRITE(LLU,20)
 20   FORMAT(" DISC LU:  ")
      READ(LLU,*)DLU
      IF(DLU.EQ.0) GO TO 9990
      IF(DLU.EQ.2.OR.DLU.EQ.3) GO TO 10
      IF(DLU.LT.0) DLU=-(DLU)
C
C   GET THE CARTRIDGE DIRECTORY AND DETERMINE IF THE DISC LU IS
C   MOUNTED.  IF SO, GET THE CARTRIDGE NUMBER AND THE FILE DIRECTORY
C   TRACK FROM THE CARTRIDGE DIRECTORY ENTRY CORRESPONDING TO THE
C   DISC LU.
      CALL FSTAT(STATUS)
      J=1
      DO 60 I=1,31
      IF(STATUS(J).NE.0) GO TO 50
      WRITE(LLU,40)
 40   FORMAT(" DISC NOT MOUNTED"/,)
      GO TO 10
 50   IF(STATUS(J).NE.DLU) GO TO 60
      J=J+1
      DRTRK1=STATUS(J)
      J=J+1
      CRNUM=STATUS(J)
      GO TO 70
 60   J=I*4+1
C
C   READ THE FILE DIRECTORY FROM THE USER'S LU.
C
 70   CALL EXEC(1,DLU,DIRBUF,16,DRTRK1,0)
C
C   OBTAIN THE # OF DIRECTORY TRACKS, AND THE # OF SECTORS/TRACK.
C
      NMDTRK=DIRBUF(9) *(-1)
      SECTRK=DIRBUF(7)
C
C
C   READ 2 SECTORS AT A TIME.
C   FIND THE NEXT 2 SECTORS TO READ BY USING THE FORMULA:
C      SECTOR ADDRESS = (BLOCK*14) MODULO (#SECTORS/TRACK)
C
C
C   TKCNTR=THE DIRECTORY TRACK COUNTER.
```

```
C  COUNTR=THE SECTOR COUNTER.
C
      TKCNTR=1
C  COUNTR=SECTOR COUNTER.
 80   COUNTR=0
 90   TEST=COUNTR*14
 100  IF(TEST.LT.SECTRK) GO TO 110
      TEST=TEST-SECTRK
      GO TO 100
 110  CONTINUE
C
C
C  READ THE 16 WORD FILE DIRECTORY ENTRY
C
C
C  FIRST DETERMINE IF PROGRAM IS TRYING TO PROCESS THE DIRECTORY TRACK
C  IF SO, SKIP TO NEXT FILE DIRECTORY ENTRY. IF A FILE CONTAINS
C  AN EXTENT, IT FIRST MUST HAVE A VALUE .GE. ZERO IN THE
C  FIRST WORD OF THE FILE DIRECTORY ENTRY, AND THEN MUST HAVE
C  AT LEAST 1 BIT SET IN THE UPPER BYTE OF WORD 5.  IF BOTH
C  ARE TRUE THEN CLEAR THE FILE'S EXTENTS.
C
C
      J=1
      DO 210 I=1,8
      CALL EXEC(1,DLU,FILDIR,128,DRTRK1,TEST)
      IF(FILDIR(J).GE.0) GO TO 120
      GO TO 190
 120  CONTINUE
      IF(FILDIR(J).EQ.0) GO TO 9990
C
C  DETERMINE IF THERE IS A BIT SET IN THE UPPER BIT OF WORD FIVE.
C
      L=J
      L=L+5
      IEXT=IAND(177400B,FILDIR(L))
      IF(IEXT.EQ.0) GO TO 190
C
C  READ NAME OF THE FILE THAT CONTAINS EXTENTS AND NOTIFY THE USER
C
      M=J+2
      K=3
      AB=1
      WRITE(LLU,130)(FILDIR(N),N=J,M)
 130  FORMAT(" CLEARING EXTENTS FOR ",3A2)
C
C  PLACE THE FILE NAMR INTO BUFFST, AND THE FILES SEC. CODE AND CR#
C   INTO THE TEMPORARY FILE'S NAMR.
C
      DO 140 N=J,M
      BUFFST(K)=FILDIR(N)
      NAMEP(AB) =FILDIR(N)
      AB=AB+1
      K=K+1
 140  CONTINUE
C
      IGUES=J+8
      CALL CNUMD(FILDIR(IGUES),IARRAY)
      DO 150 X=1,3
      Y=X+6
      Z=X+18
      W=X+2
      BUFFST(Y)=IARRAY(X)
      BUFFST(Z)=IARRAY(X)
 150  CONTINUE
C
      CALL CNUMD(CRNUM,IARRAY)
      DO 160 X=1,3
      Y=X+10
      Z=X+22
      BUFFST(Y)=IARRAY(X)
      BUFFST(Z)=IARRAY(X)
 160  CONTINUE
C
C ATTEMPT TO STORE THE FILE WITH EXTENTS INTO THE TEMP FILE.
C
      CALL EXEC(23,FMGR,2HFI,2HLE,2HNM,4,0,BUFFST,29)
C
C  DETERMINE IF THE STORE WAS SUCCESSFUL.  IF NOT, ABORT THE PROGRAM.
C
      CALL OPEN(IDCBS,IERR,NAMET,0,FILDIR(IGUES),CRNUM)
      CALL CLOSE(IDCBS)
      IF(IERR.GE.0) GO TO 180
      WRITE(LLU,170)
 170  FORMAT(" ERROR WITH FILE  *ABORT**")
      GO TO 9990
C
C  PURGE THE ORIGINAL FILE
C
 180  CALL PURGE(IDCBP,IERR,NAMEP,FILDIR(IGUES),CRNUM)
C
C  NOW RENAME THE TEMP FILE TO THE ORIGINAL FILE NAME.
C
      CALL NAMF(IDCBT,IERR,NAMET,NAMEP,FILDIR(IGUES),CRNUM)
C
 190  J=J+16
 210  CONTINUE
C
C  DETERMINE IF THE PRESENT DIRECTORY TRACK HAS BEEN COMPLETELY
C  SEARCHED.  IF SO, DETERMINE IF THERE IS ANOTHER DIRECTORY TRACK
C  AND IF SO CONTINUE WITH THE SAME PROCESS.  IF NOT GO HOME.
C
      COUNTR=COUNTR+1
      IF(COUNTR.EQ.48) GO TO 220
      GO TO 90
C
 220  CONTINUE
      TKCNTR=TKCNTR+1
      IF(TKCNTR.GT.NMDTRK) GO TO 9990
      DRTRK1=DRTRK1-1
      GO TO 80
 9990 CONTINUE
C
      END
```

A MINICOMPUTER-BASED RESOURCE SHARING DATAGRAM NETWORK

Robert R. Shatzer/HP Data Systems Division

ABSTRACT

This paper describes a minicomputer-based resource sharing network that uses datagrams as the transport medium, as well as describing the capabilities, typical applications, and basic design philosophy involved. The software architecture of a typical network node, its layered structure, and its effect on datagram routing are discussed. Pertinent data structures, techniques used for flow control, lockup prevention, contention management and error recovery are also fully described. Finally, performance data is presented to show the viability of the overall design.

INTRODUCTION

The current trend toward decentralized data processing and distributed control, combined with the low-cost availability of computational power, has resulted in a great demand to put that power where the work is to be done. It is now possible and desirable (both technically and economically) to place minicomputers at various locations on a manufacturing factory floor or at key points in a process operation. The placement of these minicomputers will thus lessen, or perhaps even eliminate, the need for a larger, more expensive mainframe. In so doing, the control, management, and data verification functions are placed in the area where the data is being collected, and thus where the feedback loop is the tightest.

However, the need generally still arises to provide some level of centralized control and coordination over the entire operation. This need includes such things as centralized program development, a centralized data base, the sharing of expensive peripherals, and the ready availability of management reports.

The interconnection of various processors to form a homogeneous network is a viable way to solve this dilemma. Hewlett-Packard has developed a minicomputer-based network, known as DS/1000, which provides a resource sharing and distributed function capability for the user, with no sacrifice in either direct localized task control or centralized coordination. The design represents a third generation development, based on the experience gained through the design and installation of over two hundred networks in the past five years.

The network provides the user with the ability to share those resources and functions which are normally available only at other points in the network. By using high-level programmatic intrinsics or operator commands, the user can access the file system, data structures, peripherals, or even other programs which are located at remote network nodes. The user interface is independent of the architecture of the network nodes, the CPU type, the operating system, the network topography, or the specific communications techniques used.

This topography-independent design and interface method is achieved by the implementation of a general nodal addressing scheme, and, within each node, a layered architecture. This architecture, similar in structure to that currently under consideration by the various international protocol standards committees, supports the transmission of data in packets, each of which contains all the necessary control, routing and sequencing information, thus avoiding some of the inefficiencies of virtual circuit establishment and clearing procedures. These packets are sometimes referred to as datagrams. Each datagram is handled by the network in an independent manner, so that each incurs minimum delay enroute. Thus, a portion of each user node in the network becomes part of a communications subnetwork and provides for the routing and forwarding of datagrams as required. This is done in an extremely efficient manner using firmware and memory management techniques so that the overhead per node is quite low. The subnetwork portion of each node also provides the necessary flow control mechanisms to prevent the lockup of network resources and to provide efficient contention management. The subnetwork additionally provides for effective error control and recovery.

Note: This article is reprinted from a paper presented by the author to a conference on Distributed Processing in Gaithersburg, Maryland in May, 1978. The conference was sponsored by the IEEE Computer Society and the National Bureau of Standards.

USER CAPABILITIES

The basic design philosophy behind the network is to provide a general purpose set of tools with which the user can meet his specific applications needs. In that regard, the network's basic task is to extend the local sphere of control to include geograhically remote locations. This is done in a generalized way so that the user is able to tailor the configuration of each network node and the overall network topography to meet the needs of the application, rather than distoring the application implementation to fit the constraints imposed by the network.

The network provides a variety of high-level user interface features, including remote file access, remote system requests, program to program communications, and remote command processing. These capabilities facilitate network resource sharing, distributed data file management, communications between user-written application programs, and the distribution of processor workloads. The layered architecture of the system provides this high level interface while effectively isolating the user from the tedium of network implementation details, such as contention management, error recovery, and link protocol. The technical design of the network will be covered in detail later in this paper.

The first of these user capabilities is Remote File Access. Remote File Access (RFA) provides the user with the ability to access named files that exist at other nodes in the network. These files may be on moving-head or flexible disc media or on tape mini-cartridges. The program intrinsics necessary to implement RFA are nearly identical to those a programmer would use to access a file in the local system. The primary differences are in the intrinsic name, which is a derivation of the standard file management package intrinsic name with the letter "D" as a prefix (e.g., READF becomes DREAD) and the inclusion of an additional parameter in the file location description array to specify the node at which the file resides.

Remote system requests allow the user to invoke system-level services at remote network nodes. These services, called DEXEC requests, include reading from and writing to peripheral devices, scheduling programs, enquiring as to program or peripheral status, and retrieving system time. Again, the intrinsics the user calls to invoke these remote services are very nearly identical to those called to invoke the same services locally. The normal EXEC intrinsic name is likewise prefixed with the letter "D" to specify the distributed nature of the call, and an additional parameter is required in the parameter string to specify the node at which the action is to take place. An extension of the remote read/write capability provides the user with the means to perform formatted FORTRAN READ/WRITE calls to remote peripheral devices.

The third major capability provided by the network is Program to Program Communications (PTOPC). This capability allows two or more user programs in different network nodes to exchange data and/or control information using high-level intrinsics. Because the receiving program can process the data prior to storing it or outputting it, this method provides a more generalized solution to many problems than either the RFA or DEXEC capabilities.

With Program-to-Program Communications, the user designates that the program initiating or opening the PTOPC link is the master and the receiving program is the slave. A single slave may have several masters or a single master may communicate with several slaves concurrently. Once the logical PTOPC link is established, the master initiates all communications with the slave and the slave responds by either accepting or rejecting the data or control information. When appropriate, the logical PTOPC connection may be broken by either the master or the slave.

The master program can send data to the slave with a PWRIT intrinsic, and receive data from the slave program with a PREAD intrinsic. Along with the data is passed a 20 word user-defined "tag field" which may be used to specify status, disposition of the data, or other information. The tag field may be passed without data by executing a PCONT request. A call to PCLOS will terminate the current PTOPC processing.

A typical PTOPC exchange might be as shown on the next page.

A fourth significant set of capabilities is provided in the form of Remote Command Processing (RCP). That is, the operator has available to him two interactive packages which allow him virtually unlimited access to the resources of the remote system. Using these packages, the user can manipulate remote files, schedule remote programs, check system status, and otherwise interact with a remote system as though his terminal were actually connected to that system.

```
            MASTER             :            SLAVE
-------------------------------:-----------------------------
                               :
  CALL POPEN (...) ----------------> CALL GET (...)
                               :
                               :   <Examine/Modify TAG Field>
                               :
         <-------------------------- CALL ACEPT (...)
         :                     :
<Examine/Modify TAG Field>     :
                               :
  CALL PWRIT (...) ----------------> CALL GET (...)
                               :
                               :   <Examine/Modify TAG Field>
                               :   <Process Data>
                               :
         <-------------------------- CALL ACEPT (...)
         :                     :
<Examine/Modify TAG Field>     :
  CALL PREAD (...) ----------------> CALL GET (...)
                               :
                               :   <Examine/Modify TAG Field>
                               :   <Gather Data>
                               :
         <-------------------------- CALL ACEPT (...)
         :                     :
<Examine/Modify TAG Field>     :
                               :
  CALL PCONT (...) ----------------> CALL GET (...)
                               :
                               :   <Examine/Modify TAG Field>
                               :
         <-------------------------- CALL ACEPT (...)
         :                     :
<Examine/Modify TAG Field>     :
                               :
  CALL PCLOS (...) ---------------------->
                               :         :
                               :     TERMINATE
                               :
```

In addition to the above four basic capabilities, the user has, at his disposal, the ability to do program preparation and system generation either remotely or within the local environment, for loading into a remote node. The loading operation itself may be initiated either locally or from a remote node.

GENERAL ARCHITECTURE

The DS/1000 network provides a high level of access to the remote network node without requiring the user to be concerned about message formats, communications line protocol, communications error handling, and other mundane tasks. The user need only be concerned with the task at hand; the layered architecture of the network provides the transparency required to isolate the user from the tedious aspects of data communications.

The network is also transaction-oriented; that is, each user request results in a request being sent form the master node to the slave node. The slave, after servicing the request, sends a reply back to the master. At this point, the transaction has been processed and control is returned to the user program or terminal.

Figure 1 describes the overall structure of a typical network node. As can be seen, the architecture is layered; the levels of layering correspond with those defined by ANSI Sub Committee X3S3, Data Communications. These layers have evolved for several reasons, among them being the need to subdivide control functions into logical groupings and make the various control levels functionally independent to allow for future enhancements and to provide data and control transparency to the upper, or user, level. [1] It should be noted here that the various levels in the network, as described below, are conceptual aids which are useful in the definition, the implementation, and the explanation of network internal functions. The user need not be aware of this layering in order to make effective use of the capabilities the network provides.

Level 1 is the physical interface level and takes into account the electrical and physical characteristics of the communications link. Several types of Level 1 interfaces are supported, as follows.

DATA RATE	ELECTRICAL INTERFACE	PHYSICAL INTERFACE
1 MBit	Current loop	Twisted, shielded pair
19.2 Kbit	RS-232C	RS-232C
2.5 MBit	Current loop	Coaxial cable

Level 2 is the data link control level; it is described as the Communications Access Method (CAM). CAM is implemented using a hybrid of software and firmware in order to gain both efficiency and high thruput. It is in this layer of the network architecture that the line protocol is implemented and the communications error control is done. The protocol that is currently used for both the 1 MBit and 19.2 KBit interfaces is bit-serial asynchronous with a longitudinal odd parity bit appended to each transmitted word. Block checking and error control is handled by the longitudinal parity bit and by a vertical and diagonal parity word being appended to the end of each block. The use of longitudinal, vertical, and diagonal parity for error control results in an undetected bit error rate of less than one in 10^{14} bits. [2,3] The protocol for the 2.5 MBit interface is Bisync, with some modifications for usage on a high-speed interface. The error control on this link is handled with the standard CRC-16 block check algorithm, implemented in hardware.

Level 3 is defined by ANSI as the network control level and is termed the Communications Management (CM) level. This level builds the request and reply message formats, and also manages flow control, contention problems, and lockup prevention. A complete user transaction is composed of an outgoing request and returning acknowledgement or reply; each message, or packet, is sent in the form of a datagram. A datagram is a self-contained packet of data which carries with it all of the necessary control and routing information which enables the network to deliver it to its ultimate destination. [4] Unlike a virtual circuit, the use of datagrams does not require call initiation and clearing procedures. Rather, each transaction is handled independently of any other. This technique adds significantly to the efficiency of the network in handling real-time applications such as instrumentation, process control, and in-plant operations management, since in these applications, individual transactions tend to be relatively independent and unrelated. When using datagrams, however, there is the possibility that messages will arrive out of sequence. This potential problem is handled at this and higher levels in the network architecture, so it is not a matter of concern for the user.

Communications Management also facilitates the implementation of nodal addressing and store and forward within the network. Each node in the network is assigned a unique 16-bit nodal address by the user at initialization time. The destination nodal address is then specified as part of the passed parameter string when a network intrinsic (e.g., RFA, DEXEC or PTOPC) is executed. The Communications Management software inserts both the local (source) and the destination nodal addresses into the datagram for use by the CM software throughout the network for determining both the outbound and return routing vectors. The destination nodal address is also used locally by CM to determine the appropriate communications channel for use in sending the request. The fact that the datagram is general in nature and that it contains both the source and destination nodal addresses, allows it to be freely forwarded through the network until it reaches the final destination. In this way, the user is not required to be aware of the network topography or the number of intervening nodes.

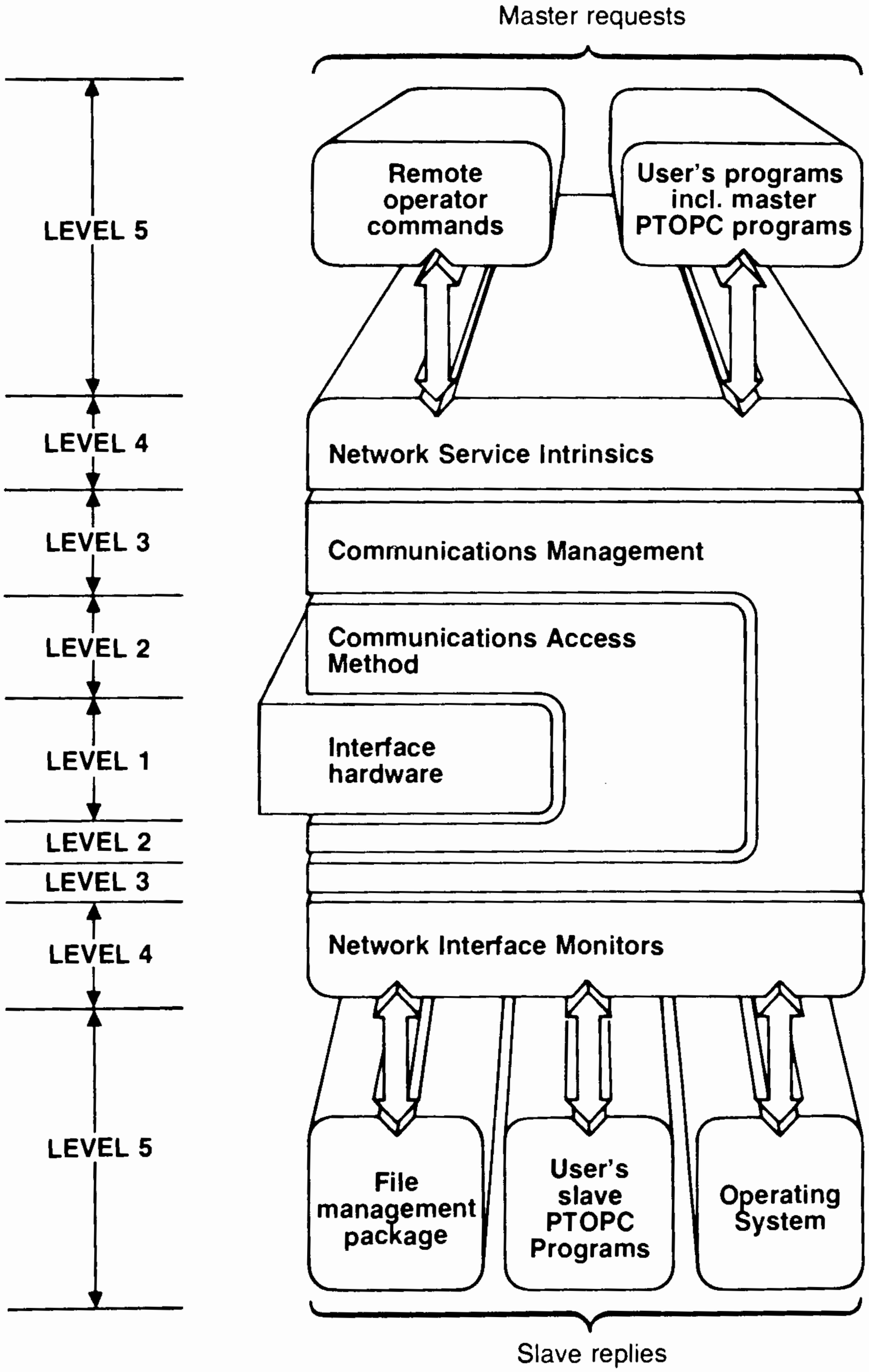

Master requests
LEVEL 5
Remote operator commands
User's programs incl. master PTOPC programs
LEVEL 4
Network Service Intrinsics
LEVEL 3
Communications Management
LEVEL 2
Communications Access Method
LEVEL 1
Interface hardware
LEVEL 2
LEVEL 3
LEVEL 4
Network Interface Monitors
LEVEL 5
File management package
User's slave PTOPC Programs
Operating System
Slave replies

Figure 1 – General Nodal Architecture

In the same manner, requests can be sent to the local node as though it were a pseudo-remote node (i.e., the destination nodal address equals the local nodal address). This feature allows the user to develop and test his programs in the local node and subsequently move them to their planned residence at another node, without any requirement for change in the program function and without any changes in the source code. This concept is termed program transportability. Transportability allows a given program to be moved, or transported, about the network and to continue to function in the same way and to interact with the same specific resources, without regard for the location of the node in which it is currently executing.

Level 4 is the system control level and contains both the Network Service Intrinsics (NSI) and the Network Interface Monitors (NIM). The Network Service Intrinsics are those user-callable subroutines which provide the user with high level access to all of the resources of the network. Those intrinsics which support Remote File Access, Distributed EXEC calls, and Program to Program Communications are typical Network Service Intrinsics. The Network Service Intrinsics support a simplified and general interface between the user and the complexities of managing network communications.

Also at Level 4, the Network Interface Monitors provide the interface between the network and the local file management package, operating system, or the user's PTOPC slave program. The appropriate NIM services that request by reconstituting it into a pseudo-local request and submitting it to a local resource manager, such as the file management package. Thus, the difference between NSI and NIM resolves to that of initiating the request (NSI), and actual performance of the requested task by the NIM.

Level 5 is the user control level. This level of the network architecture specifically addresses the end-to-end protocol necessary to fulfill the needs of the application. On the master side, Level 5 would contain the user's application programs or, alternately, the network-supported operator interface packages and the operator himself. On the slave side, this level consists of the local resource managers and the actual resources themselves, i.e., the file management package and the user's data file. For PTOPC, Level 5 would include the user's slave program and the local resources that it uses. In short, the specific definition of the functionality and composition of Level 5 is a user responsibility and is determined by the requirements of the specific application.

DATA AND CONTROL FLOW

Figure 2 details the flow of data and control within a typical network node. For the sake of clarity, only those modules and functions which are necessary to process either the master request or the slave response are shown. In the actual implementation, a given node supports both the master and the slave functions concurrently. The boundry between the two nodes and the intervening communications link is represented by the vertical broken line in the center of the figure. In the following description, the parenthesized numbers refer to the control and data flow paths shown in the figure.

Initially, the user program, in the upper left of the figure, executes a master subroutine call to a Network Service Intrinsic (1). For example, this call may be a Remote File Access file write call or a Program to Program Communication call, as discussed earlier. The user program is then suspended by the operating system until a reply is received from the remote node. The Network Service Intrinsic parses the request, checks it for errors and then builds a message block structure which will eventually be sent over the communications line. The message block, containing both the user's request and the data, if appropriate for the type of request, is then forwarded to the Communications Management routines (2). Here, the source and destination nodal addresses, a sequence number, and other control information are added to the message block, forming the request datagram. Typical request and reply datagram formats for PTOPC are shown in Figure 3. At this point, a master Transaction Control Block (TCB) is built (3). The TCB contains a master class number, a sequence number, the ID segment address of the master program, and a countdown timer which is used to return control to the user program in the event of a failure in the slave node, or on the communications link, which prevents the reply from being received. The class number and the ID segment address serve to identify the master program and specify the key to be used to rechedule it when the reply is finally received. The sequence number is used to relate the TCB to the corresponding datagram. It is possible, at any given moment, for several programs in a given node to be waiting for replies from other nodes. There is no requirement that the replies arrive in the order the requests were sent. The TCB is the means by which Communications Management software can locate the proper receiver for each incoming reply. The TCB structure is shown in Figure 4. The physical communication path is defined by a system Logical Unit number (LU). The correct LU is derived by referencing the Nodal Route Vector (NRV) table

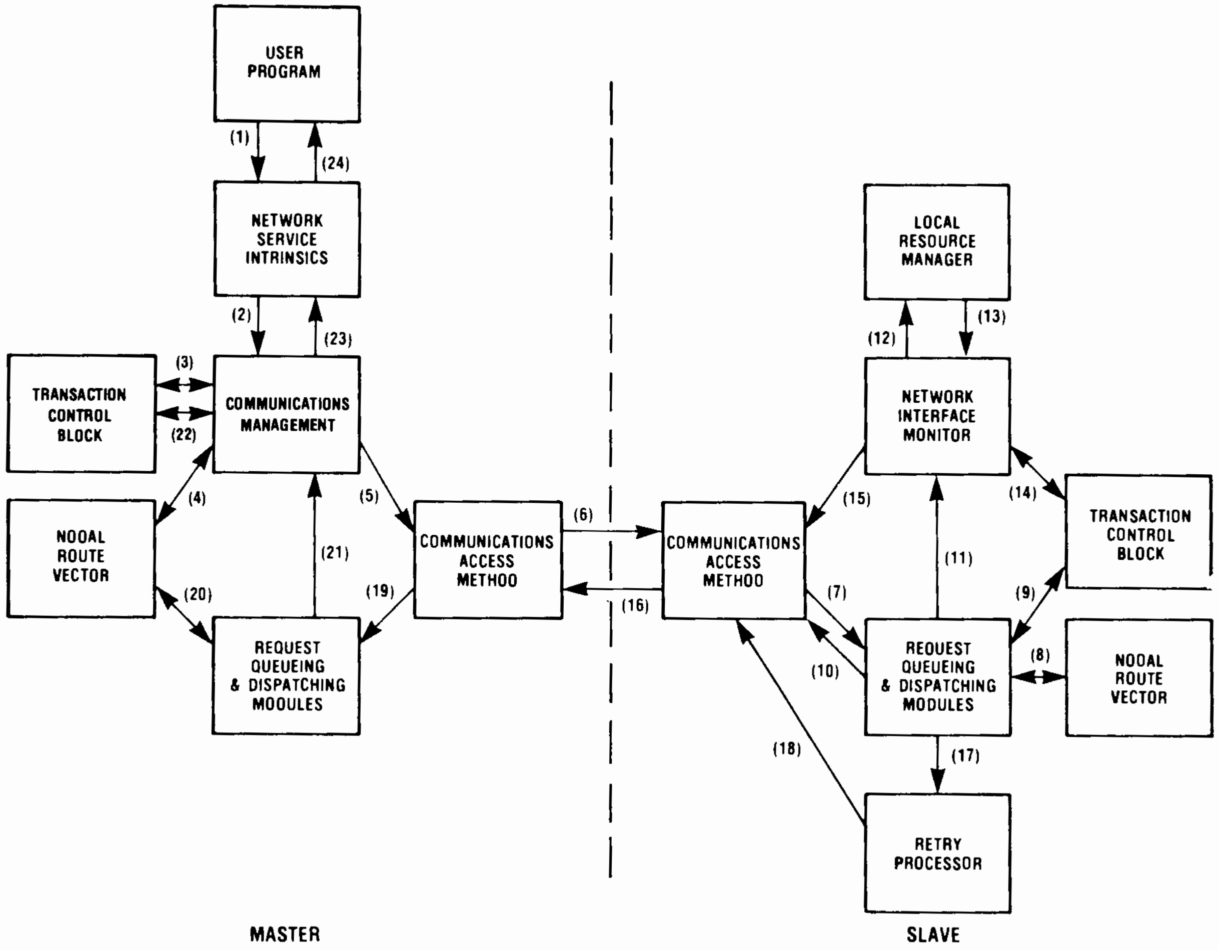

USER PROGRAM
(1)
(24)
NETWORK SERVICE INTRINSICS
(2)
(23)
TRANSACTION CONTROL BLOCK
(3)
(22)
COMMUNICATIONS MANAGEMENT
NODAL ROUTE VECTOR
(4)
(5)
(21)
(20)
(19)
COMMUNICATIONS ACCESS METHOD
REQUEST QUEUEING & DISPATCHING MODULES
(6)
(16)
COMMUNICATIONS ACCESS METHOD
LOCAL RESOURCE MANAGER
(12)
(13)
NETWORK INTERFACE MONITOR
(15)
(11)
(14)
TRANSACTION CONTROL BLOCK
(7)
(9)
(10)
REQUEST QUEUEING & DISPATCHING MODULES
(8)
NODAL ROUTE VECTOR
(18)
(17)
RETRY PROCESSOR
MASTER
SLAVE

Figure 2 – Control and Data Flow

with the destination nodal address (4). The NRV is a node specific table which defines, for each node in the network, that specific LU which the local node must use in order to access the remote node. The structure of a typical Nodal Route Vector and the manner in which it is derived from the Network Description Table is shown in Figure 5.

Following construction of the TCB, a call is made to the Communications Access Method, which consists of the communications driver and its supporting firmware (5). The driver microcode handles the actual transmission of the request and the data to the remote node (6). The entire transmission block, both request and data, is error checked by using vertical, longitudinal and diagonal parity, yielding error control better than that provided by CRC-16. Error retries are handled by the firmware, interacting with the software driver, providing a very low system overhead. Note here that, although the Transaction Control Block, the Nodal Route Vector, and the Request Queueing and Dispatching Modules are shown separately for clarity, they are actually part of Level 3 Communications Management.

At this point, the request datagram has been sent from the master node to the slave node. The first portion of the request is received by the slave node's Communications Access Method as an unsolicited interrupt. The receipt of this interrupt causes the Request Queueing Module to initiate a read request to the communications driver (7). The request datagram is then transferred to System Available Memory by the communications driver firmware. The Request Dispatching Module examines the destination information contained in the datagram, referring to the local NRV for this particular node (8). If the destination

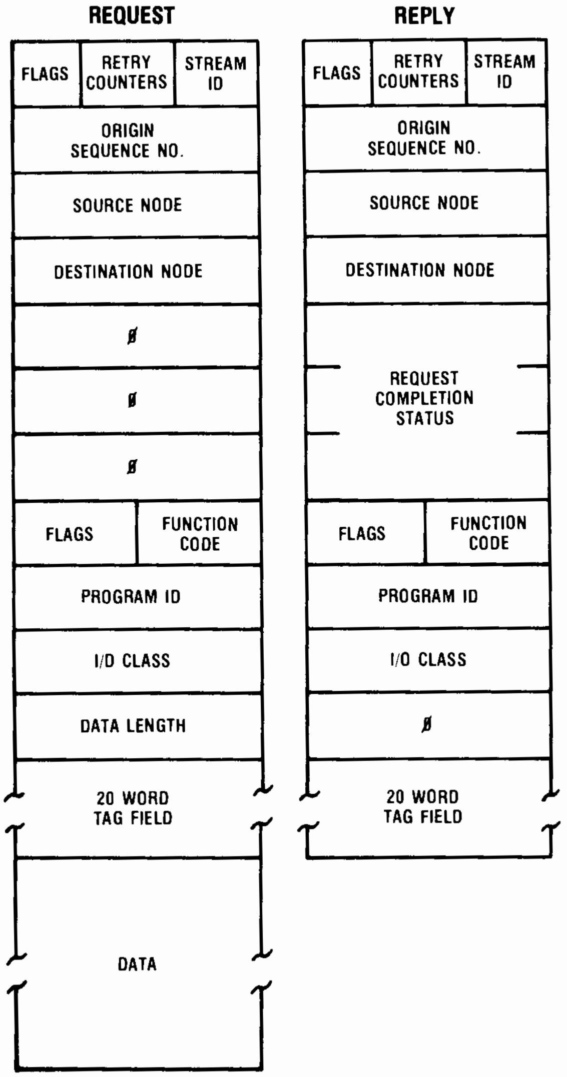


Figure 3 - Typical Datagram Formats

nodal address in the datagram matches the local node number, a slave TCB is built (9). This TCB contains information similar to that contained in the master TCB, except that a local sequence number is assigned for local reference purposes. In addition, it contains the identity of the Network Interface Monitor to which the request is to be passed for its ultimate servicing. The slave TCB also contains a timer which is used to clean-up and de-allocate system resources in the event the Network Interface Monitor cannot complete the servicing of the request. (Refer to Figure 4.)

In the event the destination nodal address in the datagram does not match the local node number, the NRV is referenced to determine the proper I/O logical channel to be used for forwarding the message to its destination. A new I/O request is then issued to pass the request datagram on toward its proper destination (10). In this way, the memory buffered store and forward function is accomplished. As can be seen, store and forward is handled with very low system overhead and is general in nature, such that the request will automatically propagate from node to node, along the route specified by the Network Description Table, until it reaches the final destination node.

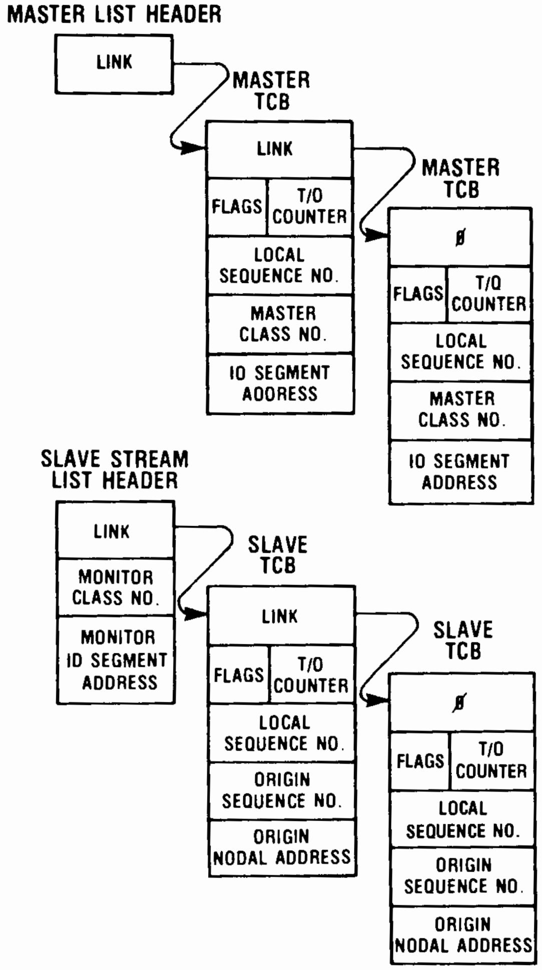
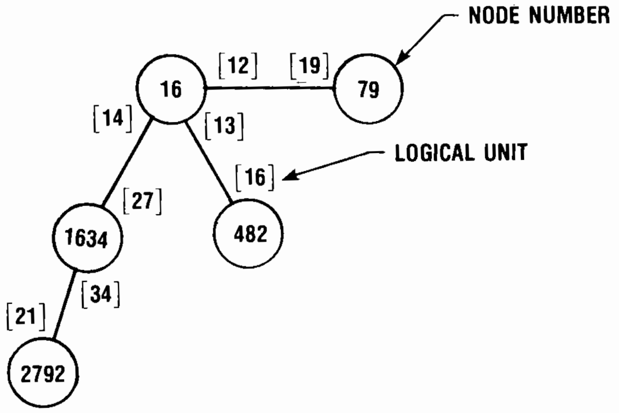


Figure 4 – Transaction Control Block Formats

After the slave TCB is built, the request and data are passed to the specific Network Interface Monitor (11), which reconstructs the particular user call and then executes it by invoking the local resource manager (12). For example, if a DWRIT call were to be issued by the master program to write a record to a remote file, at the slave node, the call would be converted into a local file management package WRITF call which would access a specific local file. It is here that the actual Data Control Block for the file is maintained by the Remote File Access Monitor. In general, most flags and control fields are maintained on the slave side so that only a minimum of control information need be transmitted over the communications line. In the case of Program to Program Communications, the local resource manager is actually the user's slave PTOPC program. The interface between that program and the PTOPC Network Interface Monitor exists at a high level (as shown earlier), and is managed by the NIM.

When the user request has been serviced by the Local Resource Manager, the local request completion information (and data, if appropriate to the type of request) is gathered (13) and a reply message block is built by the NIM. The reply contains the same basic information as the initial request message block but it also contains a flag to indicate that it is a reply instead of a request. In addition, it contains the request completion status information and any error codes that may have been generated. Since the routing information is still contained in the message block, it is, in fact, the reply datagram. (Refer to Figure 3.)

A TYPICAL NETWORK:



THE NETWORK DESCRIPTION TABLE: (NDT)

"A GENERAL TABLE STRUCTURE THAT DEFINES THE TOPOGRAPHY OF THE ENTIRE NETWORK."

THE NODAL ROUTE VECTOR: (NRV)

"A HORIZONTAL (ROW) ELEMENT FROM THE NDT WHICH SPECIFIES, FOR A GIVEN NODE, THE LOCAL UNIT IT USES TO ACCESS THE REMOTE NODES."

USE THIS LOGICAL UNIT

FROM THIS NODE → TO THIS NODE	1634	16	482	79	2792
1634	—	27	27	27	34
16	14	—	13	12	14
482	16	16	—	16	16
79	19	19	19	—	19
2792	21	21	21	21	—

NRV FOR NODE 16:

Figure 5 – Generalized Nodal Addressing

At this point, the slave TCB is no longer needed, so it is deallocated (14) and the reply datagram is submitted to the Communications Access Method (15), via a portion of Communications Management (not shown) that is appended to the NIM, to be sent back to the master node (16). Once again, the driver firmware and software take care of sending the datagram and handling the retries in the event of a communications error. When the datagram is given to CAM, the Network Interface Monitor is then free to return to its request queue and begin servicing any other request which may have arrived while it was servicing the current one. This will happen even if there are I/O transactions currently pending on the communications channel. In this way, the Monitor has maximum availability for servicing user requests and is not suspended during I/O wait time. Since there is a Network Interface Monitor for each major network function that is supported (RFA, DEXEC, PTOPC, etc.), system queue waiting times are reduced due to the use of multiple single-server queues.

When the reply datagram has been sent by the Communications Access Method, the I/O completion status is returned to the Dispatching and Queueing Modules (7). If there were no communications errors, the I/O completion status is discarded. If an error was returned to this level, the reply datagram will be re-transmitted a specified number of times via the Retry Processor (17). The Retry Processor will resubmit the datagram to the Communications Access Method for re-transmission (18), after a time delay to allow adverse communications line conditions a chance to clear. If the transmission is successful this time, the I/O completion information (7) is discarded by the Dispatching and Queueing Modules and the memory used by the datagram is released. If not, the cycle is repeated, via the Retry Processor, until it is successful or the retry count in the datagram is exhausted, at which point a diagnostic message is logged on the system console of the local node. This procedure is followed in order to relieve the Network Interface Monitor of the burden of handling communications line failure problems that would make it unavailable to service other requests. It should be noted here, as on the master side, the TCB, the NRV, the Request Queueing and Dispatching Modules and the Retry Processor actually comprise the Level 3 Communications Management structure. They are shown as separate modules for clarity.

When the reply datagram is received at the master node, it is again processed as an unsolicited interrupt to the Communications Access Method. The driver, upon receiving this interrupt, schedules the Request Queueing Module which, in turn, initiates a read request to the communications driver to read the reply datagram (19). Once again, the reply datagram is transferred to System Available Memory. The Request Dispatching Module examines the destination information contained within the datagram and compares it to the NRV to determine if this is the originating, or source, node (20). Note here that the sense of the source and destination nodes is reversed, because the reply flag is set in the datagram; "source" is always the master requestor and "destination" is always the slave node. If the local node is not the originating node, the request is forwarded via the appropriate LU as described above.

The reply datagram is now returned to the master node Communications Management routines (21) where the communications header information is stripped off and the master TCB is deallocated (22). Control is then passed through to the Network Service Intrinsic (23) where the reply information is reformatted to match the information type and format that would have been returned if the corresponding call were executed locally. Finally, control is passed back to the user program, where execution is resumed (24). The DS/1000 network transaction is complete at this point.

If the user specifies that the destination node is local, i.e., the request is to be pseudo-remote as described above, the same general flow is followed, except that the communications driver, and hence the Retry Processor, is bypassed. That is, the request datagram on path (5) feeds directly into path (7) and the reply datagram on path (15) returns via path (19) and all modules would, of course, be contained within the same node.

In addition to the modules and data flow paths described above, there is a routine to handle the diagnostic error reporting that may be necessitated by error conditions detected within the Request Queueing and Dispatching Modules. This is done so these critical response time modules are not impeded by the effort or overhead involved in reporting these messages. Another module performs the critically important watchdog timer function for all master and slave Transaction Control Blocks and, in general, deallocates system resources when they are left locked by an abortive communications error. Various other support programs, such as a node initialization and start-up routine, and an interactive Network Description Table generator, are also supplied with the network software.

FLOW CONTROL AND ERROR RECOVERY

Flow control in the network is implemented by several different methods. First, when the user initializes each node, he must specify the maximum number of Transaction Control Blocks that are allowed to exist at that node at any one time. From this input, a pool of available TCBs is created; this pool is used for both master and slave TCBs. For ease of management, each TCB is threaded into one of several linked lists as shown in Figure 4, depending on whether it is null (available), master, or supporting a particular slave function. In usage, the TCBs migrate between the null list and one of the active lists. When a master request is made, and there are no null TCBs available, the master program is suspended until one becomes available. When an incoming slave datagram is received, the Request Queueing and Dispatching Modules request the use of a TCB. If none is available, the request is returned to its originator, or to the previous intermediate node, with a busy reject condition indicated in the datagram. After a short while, the master Communications Management software will attempt to re-issue the request. This process is repeated until either the master TCB times out or a retry counter is exhausted, at which time the appropriate error codes will be returned to the user level on the master side. All TCBs have integral timers in them, to avoid the potential lockup condition brought about by TCB users failing to deallocate them.

A similar type of flow control is exercised by the potential unavailability of System Available Memory (SAM). Since SAM is a finite resource, and all incoming and outgoing requests pass through it, automatic flow control results as a function of the availability of a contiguous segment of SAM in which to place the datagram. When a master request is made and there is inadequate SAM available, the user program is suspended until the required memory becomes available. Incoming slave requests are rejected at the Communications Access Method level until sufficient memory is available to hold the datagram. Lockup is averted by having all SAM buffers keyed to TCBs so that they are timed out and deallocated in the same manner.

The above two methods of flow control work by enforcing admission delay; that is, a request is not allowed to enter the network, or the network node, unless sufficient resources exist to support it. A third type of flow control is referred to as transit delay, and is exerted at many places in the network in the form of single-server queues. Actually, as can be seen from Figure 2, the network node contains many single-server queues, ranging from the Request Queueing and Dispatching Modules to the peripheral devices themselves.

Error control and recovery is handled at several levels. First, at Level 2, the Communications Access Method driver and microcode repeat certain elements of line protocol or retransmit entire datagrams, as necessary, until a specified retry count is reached. Retransmission requirements are determined by the CAM microcode upon detection of either a protocol failure or an incorrect block check sequence. This level of error control is adequate to handle the classical communications line error problems.

Further error control takes place at the next Level 3, where the Communications Management software supports several techniques. If, at any node, CAM is unable to handle the classic communications line error conditions described above, control is returned to this level. Communications Management determines whether its retry count for the particular error has been exhausted. If not, a delay value appropriate to the particular error type is calculated, the retry counter is incremented, and the datagram is handed to the Retry module. Retry will re-submit the datagram for transmission after this delay. If, on the other hand, the retry count has been exhausted, the error indication for that datagram is sent to the Network Service Intrinsic level at the originating node, in the case of a master request, or, alternatively, the datagram is discarded, and a message indicating the cause, source and destination node numbers, and other information is displayed on the system console of that node; in this latter case, the program at the originating node which made the master request will receive a time-out error message. When the error condition at the originating node results from insufficient or busy resources at the remote node, the Network Service Intrinsic will re-submit the master request at intervals that can be specified by the person managing the network. In this way, Communications Management is able to handle a wide variety of network problems including those imposed by temporary or permanent resource constraints at a node.

IMPLEMENTATION

The network is implemented using Hewlett-Packard 1000 computer systems. Each node in the network contains a Hewlett-Packard Real-Time Executive operating system (RTE) with 32K or more words of memory. Any node in the network may be either disc or memory based. The disc-based nodes support 20 and 50 MByte moving-head drives as system discs, while the memory based nodes can support a flexible disc or 3M-type tape mini-cartridge as either a peripheral device or file system resource.

Figure 6 shows typical throughput data rates on a single communications link for a star network with from one to five satellites. The communications link is the 1 MBit current-loop interface. Figure 7 shows the same data for a full-duplex 9.6 KBit modem link. Both of these sets of curves show the throughput as a function of buffer size for Program to Program Communications transfers.

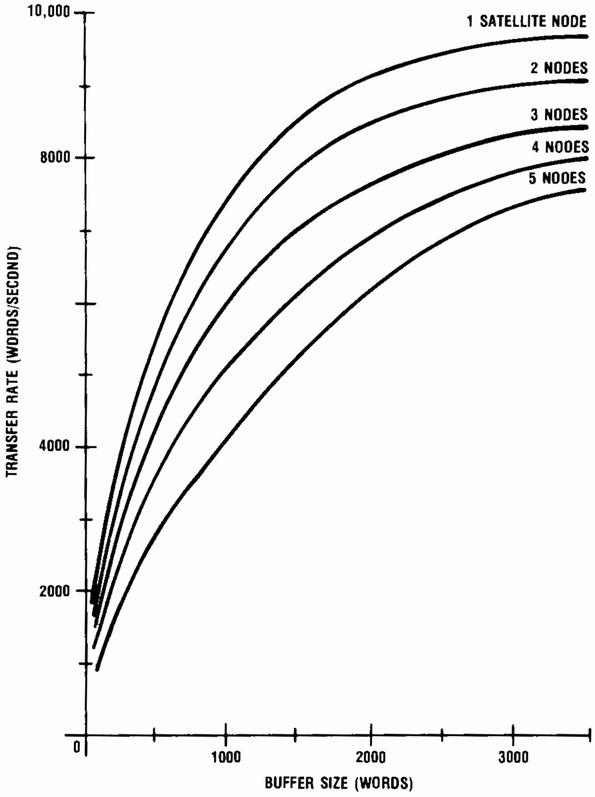


Figure 6 — Star Network (PTOPC Transfers @ 1 Mbps)

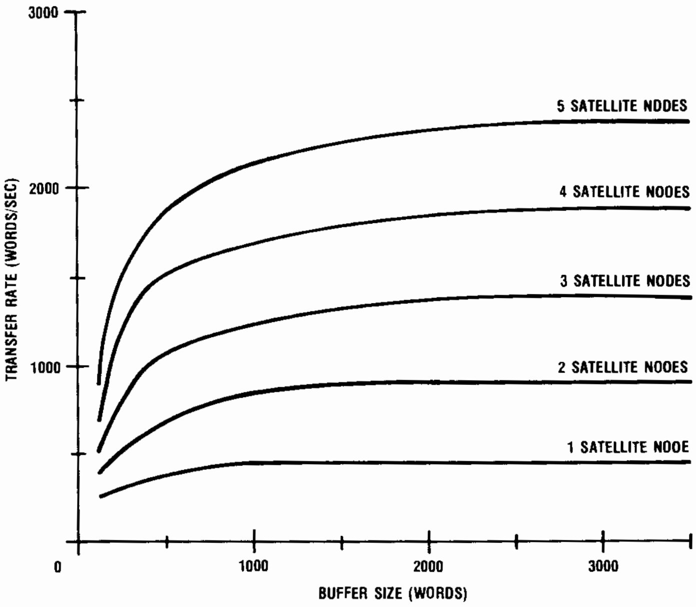


Figure 7 — Star Network (PTOPC Transfers @ 9.6 Kbps)

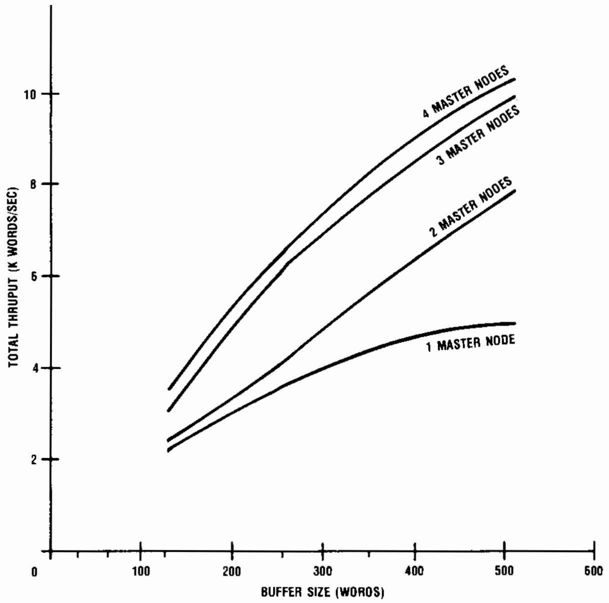


Figure 8 — Star Network (RFA Transfers @ 1 Mbps)

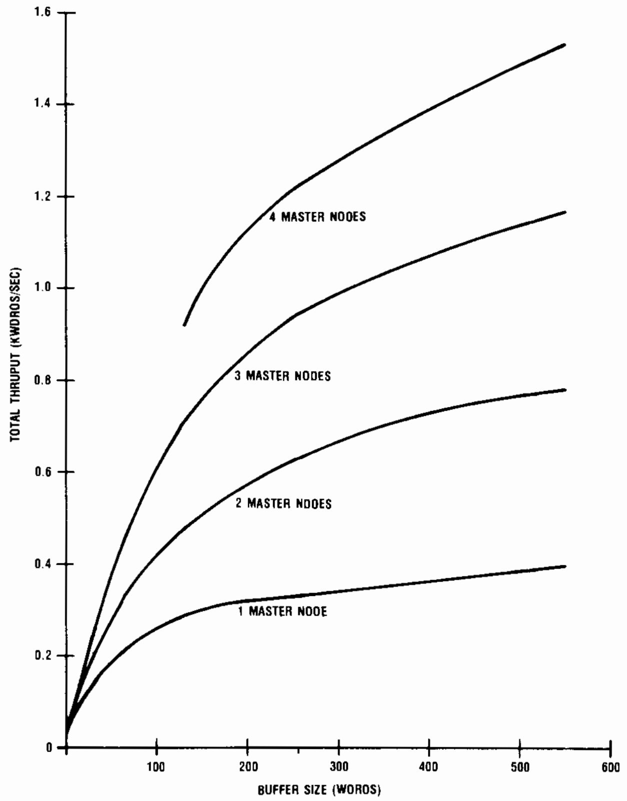


Figure 9 — Star Network (RFA Transfers @ 9.6 Kbs)

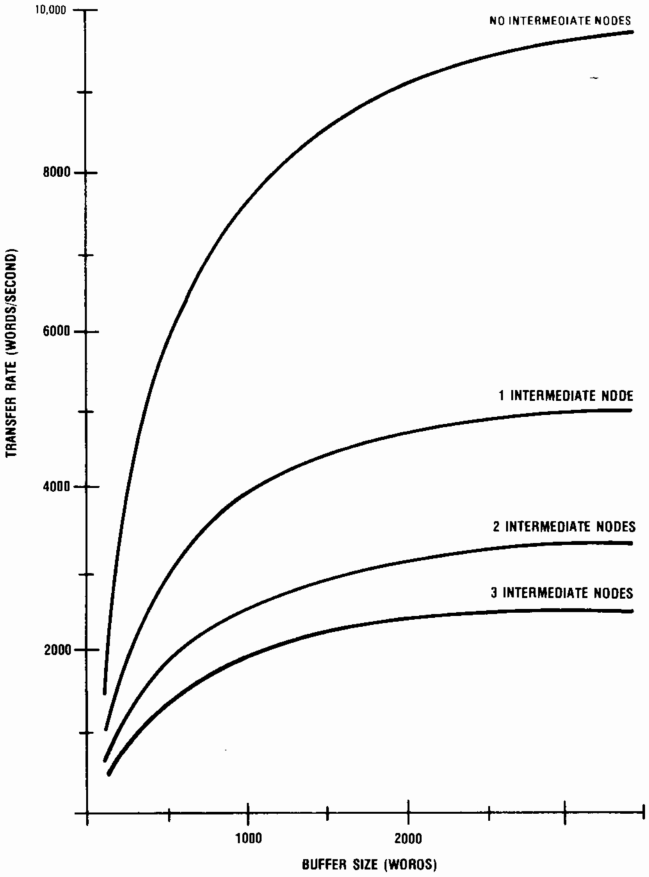


Figure 10 — String Network (PTOPC Transfers @ 1 Mbps)

Figures 8 and 9 show the same type of data for the 1 MBit and 9.6 KBit links, respectively, but for Remote File Access requests. Here the data rate is significantly less due to disc controller, file directory, and data control block contention. Note that the data is plotted as total (aggregate) thruput vice thruput on a per-node basis, as are Figures 6 and 7.

Figure 10 shows the effect of store and forward delays on a string network as a function of both the buffer size and the level of hierarchy within the data path. While this curve shows the effect of store and forward delay on PTOPC transactions, the data is equally valid for any other type of transaction, since the intermediate nodes do not evaluate the request type.

References:

1. Emmons, W. F., "Data Network Protocol Standards," *Proceedings of Computer Networking Symposium,* December 15, 1977, IEEE Computer Society (77CH1252-6 C).

2. Rowland, R., "Error-detecting Capabilities of Two Coordinate Parity Codes," *Electronic Engineering,* January 1968.

3. Halkon, R., "Error-detecting Capabilities of Parity Codes," *Electronic Engineering,* July, 1969.

4. Folts, H. C., "Interface Standards for Public Data Networks," *Proceedings of Computer Networking Symposium,* December 15, 1977, IEEE Computer Society (77CH1252-6 C).

UNDERSTANDING RJE/1000

Robert Gudz/HP Data Systems Division

Your HP 1000 system can be linked to an IBM System 360/370 mainframe once you have installed the RJE/1000 subsystem. With the Remote Job Entry capability it provides, many new opportunities become available to increase the usefulness of your RTE system. RJE/1000 is a versatile package for communicating data between your RTE system and an IBM mainframe host. Data transmission employs the format conventions and protocol of IBM's BISYNC protocol. This article is intended to build a framework of understanding by discussing these and other fundamental principles of RJE/1000's operation. Also, a practical guide to help you get started using RJE/1000 is given in order to eliminate the frustrations that commonly arise from the mysteries of datacomm.

What is RJE/1000?

Remote job entry began as a means of extending the power of a central computing site by allowing jobs to be submitted and run from remote stations. The IBM 2780 has been the most popular workstation. It consists of a cardreader, lineprinter, and cardpunch. Because the device has been supported by all major IBM operating systems since it was introduced in 1967, it has become a widely emulated standard for providing a remote batch capability. RJE/1000 is a software/hardware subsystem for the HP 1000computer which emulates the 2780. That is, as far as the IBM 360/370 mainframe host is concerned, a remote HP 1000running RJE/1000 is no different from a 2780. However, the sources and destinations for data at the HP 1000 include devices such as tape drives and CRT terminals, as well as disc files. And the operation of the RJE/1000 subsystem can be initiated and controlled programatically, as well as by a terminal user. These capabilities allow the user to: (1) collect real-time experimental data locally at an HP 1000; (2) batch the data to an IBM 370 for analysis; and (3), retrieve the results (or, perhaps, new specifications) back to a file on the HP 1000 for use by the measurement program, etc. The data and control aspects will be discussed in more detail later.

First, let's describe the HP 1000 system environment required to build a working RJE/1000 subsystem.

1. HP 1000 M, E, or F-Series Computer, running RTE-II (64 Kb), RTE-III (96 Kb), or RTE-IV (128 Kb);
2. HP 12618A Modem Interface Kit (two cards which are included in the RJE/1000 subsystem);
3. HP 12620A Privileged Interrupt Board;
4. RJE/1000 Subsystem Software (91780A);
5. A supported modem with a dial-up or leased line connection (at up to 9600 baud) to an IBM 360/370 providing BTAM, TCAM, HASP, RES, or JES2 Access Methods.
6. The newly revised RJE/1000 Programmer's Reference Manual.***

Data

The major aspects of data flow within RJE/1000 are:

- Data sources and data destinations within RJE/1000.
- The method by which data is sent over the communication link between RJE/1000 and the IBM 370, i.e. BISYNC protocol.
- How data is converted within the HP 1000 to and from the BISYNC data on the line.

*** Remember, this article is not intended to replace the Programmer's Reference Manual!*** Instead, it is supposed to present underlying concepts and fundamentals, as well as some helpful hints.

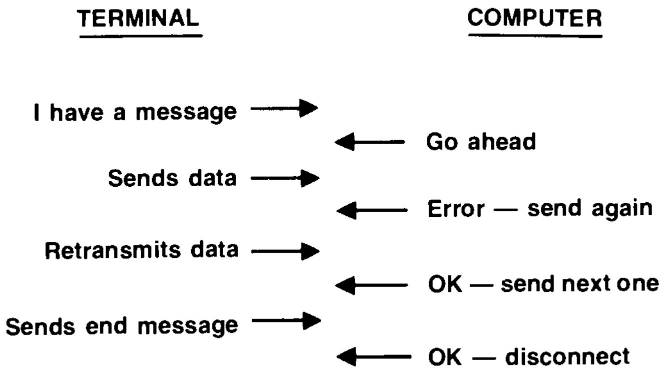
Data Sources/Data Destinations

As previously mentioned, an IBM 370 communicating with a RJE/1000 system believes it is talking to a 2780. Hence, there are three types of data flow — three data streams — which correspond to input from the card reader, output to the line printer, and output to the card punch. With an RJE/1000 system , however, the actual sources or destinations of these three data streams can be any standard devices or files. A single command may be used to change the assignment of these data streams at any time during the operation of RJE/1000. The command #C, p1, p2, p3 causes the input stream to be read from p1, the list output stream to be sent to p2, and the punch output stream to go to p3, where p1, p2, and p3 can be either the LU of a device, or a filename.

The command string which controls RJE/1000 is derived from the input stream; that is, the input stream can contain both data records for IBM and commands for RJE/1000. The RJE/1000 commands are stripped off and used to control the RJE/1000 subsystem. For example, to send a job to the 370 under interactive control at the HP 1000, one would first change the input stream to the device or file containing the job, and then have as the last record in the job an RJE/1000 command which changes the input stream back to the LU of the controlling terminal to await further commands. Input records which precede the RJE/1000 command are treated as data and are transmitted to the 370 using the BISYNC protocol discussed next. Data that may contain records that look like RJE/1000 commands (i.e., they begin with a number sign #) can be sent in a "transparent" manner.

Data Transmission — What is BISYNC?

Introduced by IBM in 1966, Binary Synchronous Communication (BISYNC, or BSC) became the defacto standard protocol for half duplex synchronous data transmission. The basic format was developed to send blocks of records corresponding to 80 column punched-card images. The data is treated as bytes and is packaged into message blocks that begin and end with special control bytes. A error checking code called CRC-16 (two bytes long) is included for each block and is recomputed by the receiver who then replies with either an ACK acknowledging it or a NAK (negative acknowledgment) requesting a retransmission. This constant interaction between sender and receiver is sometimes referred to as "handshaking".



About a dozen special characters were reserved as control characters in BISYNC and must not be included in the data. These control characters are listed here along with their representations in both IBM's EBCDIC and the USASCII character codes.

BSC CONTROL CHARACTERS USED WITH RJE/1000

RJE can receive all characters, but does not ordinarily send SOH or RVI.

DATA LINK CHARACTER	EBCDIC	USASCII	MEANING
SYN	SYN	SYN	Synchronous Idle — Used to establish and maintain synchronization.
SOH	SOH	SOH	Start of Heading — Precedes a block of heading characters. Received but not transmitted by RJE.
STX	STX	STX	Start of Text — Precedes a block of text characters.
ETB	EOB (ETB)	ETB	End of Transmission Block — Indicates the end of a block of characters begun with SOH or STX.
ETX	ETX	ETX	End of Text — Terminates a block of characters begun with STX or SOH and transmitted as an entity. It requires a reply indicating receiving station status.
EOT	EOT	EOT	End of Transmission — Can also be used to signal an abort. Causes reset of all terminals on the line.
ENQ	ENQ	ENQ	Enquiry — Bid for line, or used to ask for retransmission of a response.
*ACK0	DLE ‘70’	DLE 0	Affirmative Acknowledgement — ACK0 — Positive response to line bid.
*ACK1	DLE/	DLE 1	ACK0/ACK1 — In sequence indicate previous block accepted with no errors and receiver ready to accept next block.
NAK	NAK	NAK	Negative Acknowledgement — Previous block received in error and receiver ready to receive retransmission.
DLE	DLE	DLE	Data Link Escape — Used only in conjunction with other control characters and transparent mode control characters.
ITB	IUS	US	End of Intermediate Transmission Block — Divides a message (heading or text) for error checking without causing a transmission reversal.
*WACK	DLE,	DLE;	Wait-Before-Transmit Positive Acknowledgement — Receiving terminal temporarily not ready to receive. Can be used as positive response to received data block or line bid.
*RVI	DLE@	DLE <	Reverse Interrupt — Positive response, used instead of ACK or ACK1.
*TTD	STX ENQ	STX ENQ	Temporary Text Delay — Sending terminal in message transfer state and wishes to retain line but not ready to transmit.
*DLE EOT	DLE EOT	DLE EOT	Disconnect Sequence — Transmitted on a switched line to indicate end of transmission.
*BCC	XXX XXX	XXX XXX	Block Check Characters — Two bytes which represent a checksum for the preceding block of data.

*Two-character sequence

Figure 2

Note that some of the control sequences consist of two characters. For example, in EBCDIC, ACK0 is really the two character sequence DLE (octal 20) 70 hex (or octal 160). ACK1 is similarly represented by the two character sequence DLE /.

The two different ACKs are used in an alternating manner in order to keep the sender and receiver in step.

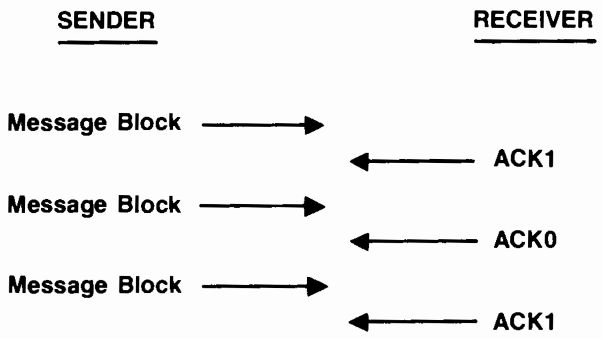


Figure 3

If the sender fails to get back an acknowledgment for the last block of data it sent, either the receiver never got the block, or else the receiver did get it, but the ACK it returned didn't get through.

If an ACK is not received, the sender transmits an ENQ to the receiver. In the responses shown below, the sender has not received an ACK1 after the second message block was transmitted. After a nominal three seconds receive timeout, the sender transmits an ENQ. If the receiver responds with ACK0, then the message was lost, since the receiver is still acknowledging the first message block. If the receiver responds with an ACK1, then the message was received but the acknowledgment lost. In the first case the sender would retransmit the second message block, and in the second case would continue with the transmission.

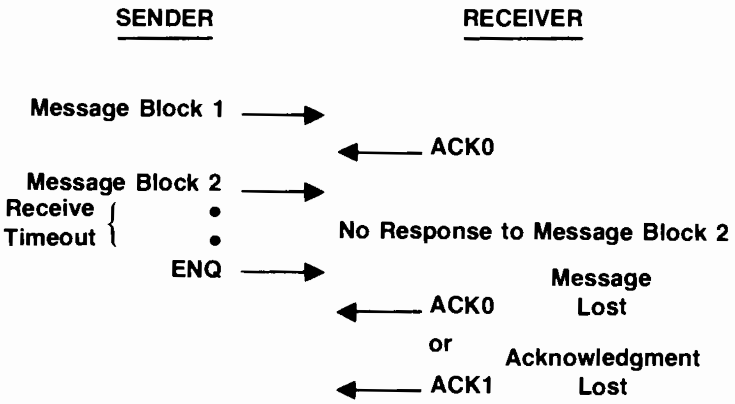
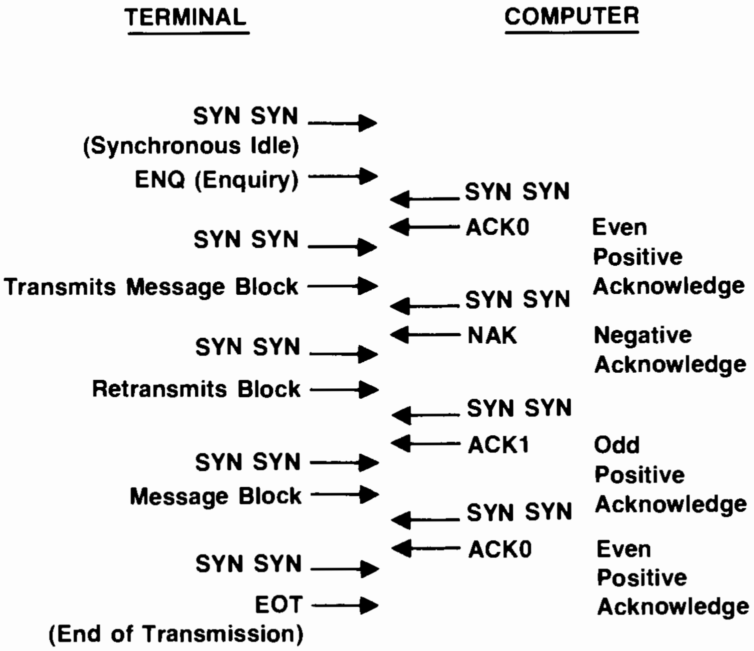


Figure 4

Each transmission sequence following a line turnaround (change in direction) begins with at least two SYN control characters in order for the receiver to obtain character synchronization from the serial bit stream. (A pad control character (often Null) is also used to account for the actual hardware turnaround delay.) The simplified handshaking sequence is now presented with the BSC control sequences actually used.



Note: Leading and trailing pads not shown.

Figure 5

To send binary data, which might very well include the special control bytes, a special "transparent" mode is entered during which all characters are permitted. The reason all data isn't sent in transparent mode to begin with is the historical, evolutionary one given above, and also because more overhead is required — additional control characters must be sent to enter and exit the transparent mode.

In cases where records may not contain all 80 characters, two options are available with RJE/1000. All short records can be automatically "padded out" to a length of 80 with Blanks, or alternatively, the records can be automatically terminated with the EM (End of Media) character. This second option reduces the volume of data transmitted and hence increases effective throughput over the data comm line. Either of the options can be specified when starting RJE/1000 by the #M "modify configuration" command.

To initiate a transmission, one side (for example, the HP 1000) must "bid" to gain control of the line. It does this by sending an ENQ character to enquire whether the other side (in this example, the 370) is willing to become the receiver. Willingness to receive is indicated by the return of an ACK0. The HP 1000 then becomes the sender and may send a message in the handshake manner described above until it terminates transmission by sending an EOT (End of Transmission) character. When this occurs, the line is once again free (or "in control mode") and either side can now bid for the line to become a sender.

Long messages are broken into a series of blocks for transmission, as illustrated below. Each block of text, except the last, is followed by an end of transmission block (ETB) character or an end of intermediate transmission block (ITB) character. ETB requires a response from the receiver and causes line turnaround and the BCC (Block Check Characters) to be sent and compared. ITB divides the message for error checking purposes and does not require a response from the receiver. After the first ITB, an SOH is not required before each text block. The last intermediate block is followed by an ETX or ETB character. As each intermediate block arrives, its BCC is compared with the receiver's BCC. If an error is detected in any intermediate block, no action can be taken until ETB is received, then all intermediate blocks must be retransmitted.

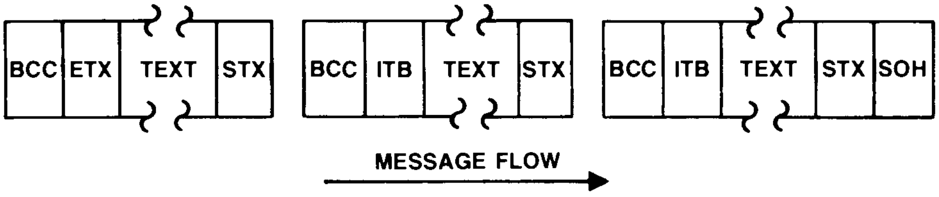


Figure 6

During transmit mode, the sender is required to send something at least every three seconds so that the receiver can be sure that nothing has gone wrong with the connection. So if the sender decides to pause between messages, it periodically sends a TTD sequence (Temporary Text Delay) to which the receiver responds with a NAK. In this manner, transmit mode can be maintained by the sender almost indefinitely between messages. A time limit (or "timeout") is set on this and other states too, though, in order to detect error situations.

Changes of transmission states, for instance from control mode to transmit mode, can readily be seen with RJE/1000 by switching on the Diagnostic Messages capability. All changes of state with respect to the BISYNC protocol are then logged on the system console using descriptive messages.

In Diagnostic Messages mode, many aspects of the BISYNC behavior of RJE/1000 become visible. Two important points should be noted.

- With RJE/1000, the change from control mode to transmit mode occurs automatically as soon as the first record is read in from the input stream. This is done by bidding for the line because RJE/1000 needs to internally move the record from its I/O buffer to its communications buffer in order to read in another record. In transmit mode RJE/1000 is the sender and the IBM/370 is the receiver. This prevents any data from coming in and overwriting the data comm buffer.

- No data is actually sent until either the data comm buffer fills up with input records (it can only hold four 80-character records) or an RJE/1000 command to send the data (and then turn the line around to listen) has been executed. In the first case, each time the data comm buffer fills while reading in a long input stream, RJE/1000 sends out its contents as a block in a long, multiblock message. A partially filled data comm buffer often remains at the end of reading in the input stream. This is sent (followed by an EOT) by any of the line turnaround commands such as #P, #E, #R, or #W.

The BISYNC protocol can be seen first-hand by using the TRACE diagnostic aid provided with RJE/1000. With TRACE enabled, all characters sent and received over the communication line can be saved (on disc or mag tape for example) for later analysis. TRACE can be enabled or disabled at any time during RJE/1000 operation to investigate specific problems. It turns out to be very useful when establishing an RJE link for the first time, because any protocol incompatibilities between the HP 1000 and the remote host are made obvious.

The saved TRACE data can then be examined using the TDUMP analysis program which is also provided with the RJE/1000 subsystem. This analysis can even be made while RJE/1000 is still running. The following sample TRACE/TDUMP listing shows the Sign-on sequence used with a remote OS/VS2 SVS system running HASPII, Version 4, Release 1. The left column of data is that sent by RJE/1000, and the right column is what was received from the remote host.

```
RJE/1000 TRACE OF  WED JUL 25, 1978   09:49:31.25
    HR:MN:SECOND  SENT          RECEIVED
NEW TRACE STARTED ***************************************
I/O REQUEST=003403   ESTABLISH MODEM CONNECTION
COMPLETION/ERROR REPORT, STATUS=002   000000
I/O REQUEST=003503   BID FOR LINE
     9:51:45.35   062  SYN
     9:51:45.35   062  SYN
     9:51:45.35   062  SYN
     9:51:45.35   062  SYN
     9:51:45.35   055  ENQ
     9:51:45.35   377  XXX
     9:51:45.35   377  XXX
     9:51:45.55                 062  SYN
     9:51:45.55                 062  SYN
     9:51:45.56                 062  SYN
     9:51:45.56                 020  DLE  DLE     }
     9:51:45.56                 160  XXX  70(HEX) } ACK0
COMPLETION/ERROR REPORT, STATUS=006   000000
     9:51:47.64   062  SYN
     9:51:47.64   062  SYN
     9:51:47.64   062  SYN
     9:51:47.64   062  SYN
     9:51:47.64   002  STX }
     9:51:47.64   055  ENQ } TTD
     9:51:47.64   377  XXX
     9:51:47.65   377  XXX
     9:51:47.84                 062  SYN
     9:51:47.84                 062  SYN
     9:51:47.84                 062  SYN
     9:51:47.85                 062  SYN
     9:51:47.85                 075  NAK
I/O REQUEST=003702   WRITE
     9:51:48. 4   062  SYN
     9:51:48. 4   062  SYN
     9:51:48. 4   062  SYN
     9:51:48. 4   002  STX
     9:51:48. 4   141     /
     9:51:48. 4   134     *
     9:51:48. 4   342     S
     9:51:48. 5   311     I
     9:51:48. 5   307     G
     9:51:48. 5   325     N
     9:51:48. 5   326     O
     9:51:48. 5   325     N
     9:51:48. 5   100
     9:51:48. 6   100
     9:51:48. 6   100
     9:51:48. 6   100
     9:51:48. 6   100
     9:51:48. 6   100
     9:51:48. 6   100
     9:51:48. 7   331     R
     9:51:48. 7   305     E
     9:51:48. 7   324     M
     9:51:48. 7   326    O
     9:51:48. 7   343    T
     9:51:48. 7   305    E
     9:51:48. 8   367    7
     9:51:48. 8   361    1
     9:51:48. 8   100
     9:51:48. 8   361    1
     9:51:48. 8   362    2
     9:51:48. 9   363    3
     9:51:48. 9   364    4
     9:51:48. 9   365    5
     9:51:48. 9   366    6
     9:51:48. 9   367    7
     9:51:48. 9   370    8
     9:51:48.10   100         although shortened for this example,
     9:51:48.17   100 <-      blanks pad out record to 80 characters.
     9:51:48.17   100
     9:51:48.18   003  ETX
     9:51:48.18   224   'M }
     9:51:48.18   157    ? } BCC (Block Check Characters)
     9:51:48.18   377  XXX
     9:51:48.18   377  XXX
     9:51:48.18   377  XXX
     9:51:48.38                062  SYN
     9:51:48.38                062  SYN
     9:51:48.38                062  SYN
     9:51:48.38                020  DLE }
     9:51:48.38                141    / } ACK1
COMPLETION/ERROR REPORT, STATUS=006   000000
I/O REQUEST=003203   SEND EOT
     9:51:48.57   062  SYN
     9:51:48.57   062  SYN
     9:51:48.57   062  SYN
     9:51:48.57   062  SYN
     9:51:48.57   067  EOT
     9:51:48.57   377  XXX
     9:51:48.58   377  XXX
COMPLETION/ERROR REPORT, STATUS=002   000000
NEW TRACE STARTED **********************************************
I/O REQUEST=003603   DISCONNECT
     9:52:39.19   062  SYN
     9:52:39.19   062  SYN
     9:52:39.19   062  SYN
     9:52:39.19   062  SYN
     9:52:39.19   020  DLE
     9:52:39.19   067  EOT
     9:52:39.19   377  XXX
     9:52:39.20   377  XXX
COMPLETION/ERROR REPORT, STATUS=000   002000
TDUMP COMPLETED WITH   0 OVERRUN ERRORS!
```

Data Translation

RJE/1000 allows the data sent over the communications link to conform to either the EBCDIC or USASCII character codes. The #M "Modify Configuration" command can be used to specify the code which RJE/1000 will use for the line. Data is then automatically translated between the internal ASCII format and the specified line code while sending and receiving.

The representations of the BISYNC control sequences in each of these codes has already been shown. Because IBM systems use EBCDIC, the TDUMP program was configured to translate the bytes in the TRACE file into EBCDIC characters. In the TRACE/TDUMP example given above, EBCDIC characters are shown alongside the octal representations of the TRACE data bytes.

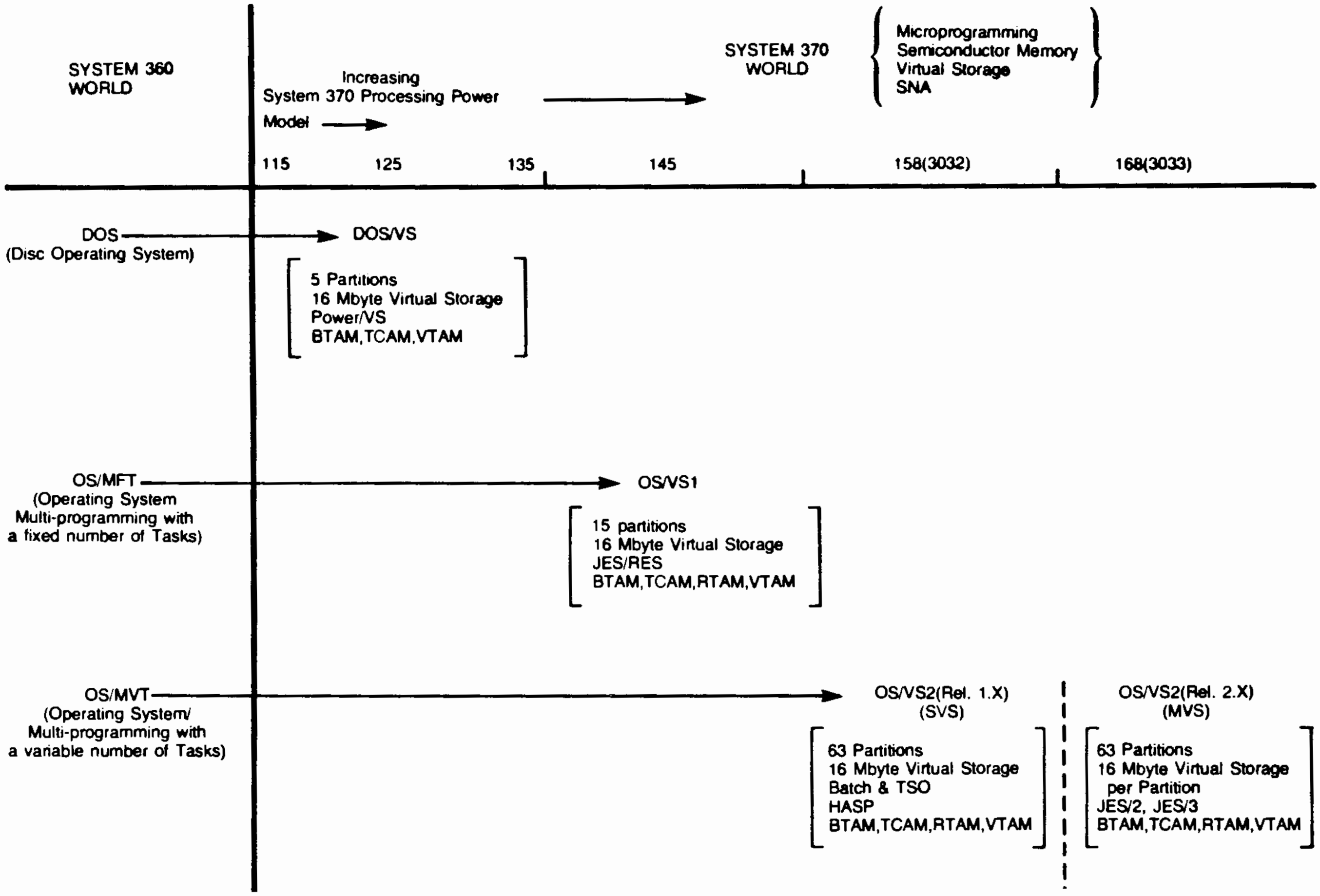
Getting Started Using RJE/1000

Once you have a RJE/1000 subsystem generated into your RTE system and have the interface cards plugged in and cabled to your modem, you are all set to start an RJE session. Well, almost all set. To make life easier, first find the answers to the following questions.

1. What operating system and remote access method does your host IBM mainframe use?
2. Has the DP center correctly configured the remote line assigned to you?
3. What is the format of the sign-on card?
4. What is the format of your job card and do you need to explicitly start your remote reader, printer, and punch processes?

The major operating systems used on IBM 360/370 mainframes are shown below.

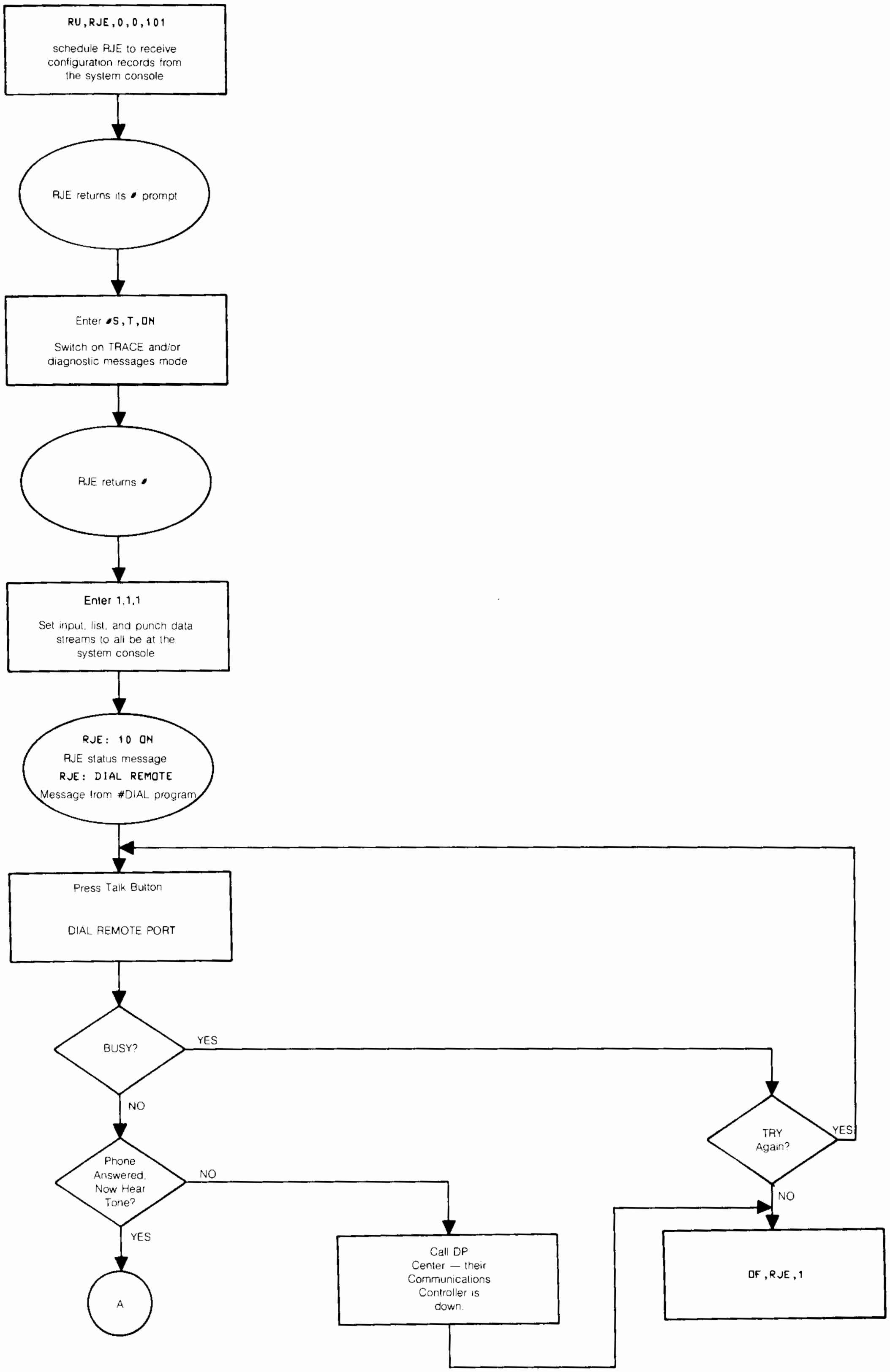
IBM MAINFRAME SOFTWARE/HARDWARE SUMMARY

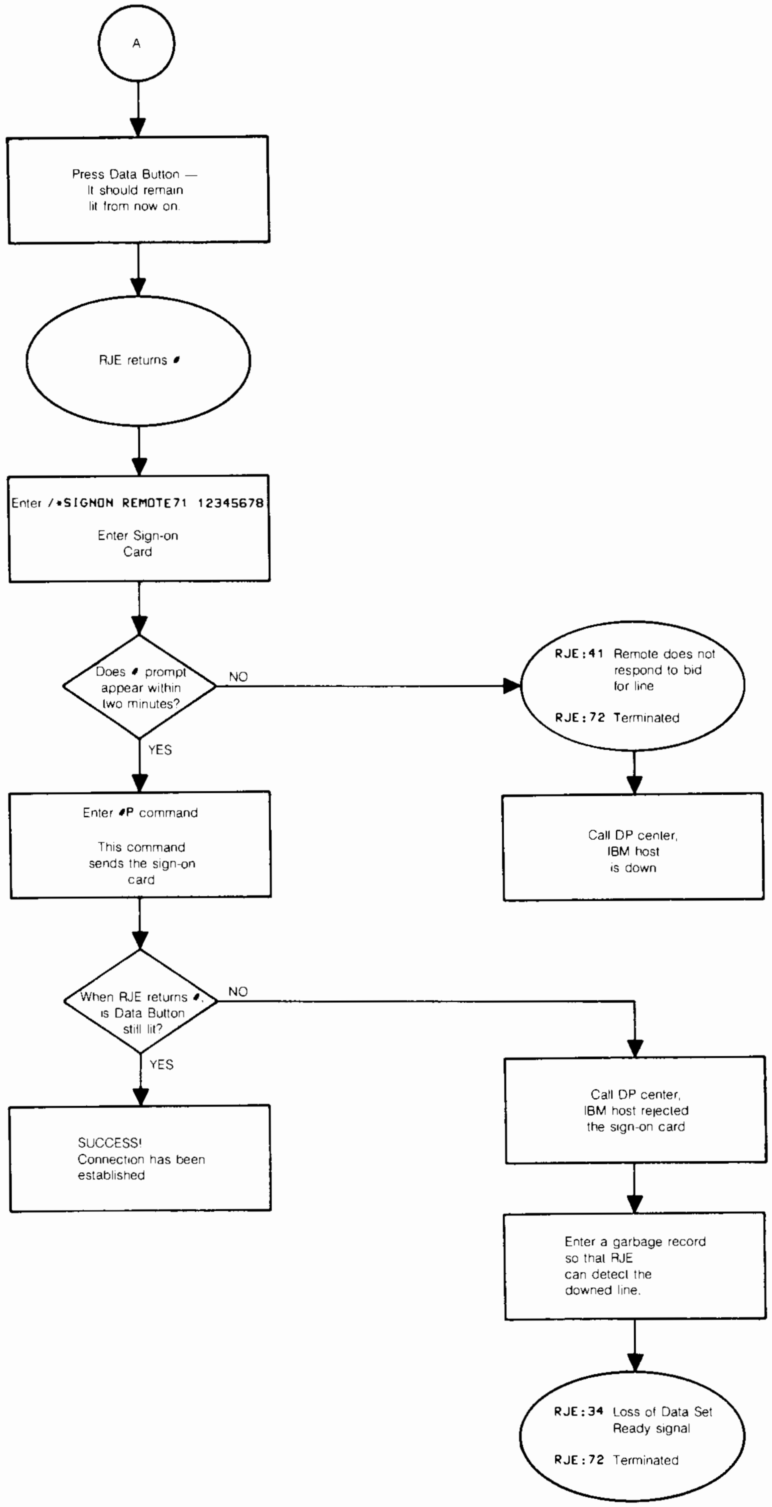


RJE/1000 currently supports use with the RTAM, BTAM, and TCAM access methods, as well as remote job entry to the HASP, RES, and JES2 job schedulers. The job schedulers each have their own set of commands which should be obtained from your DP center.

The following flowchart should be used to initiate an RJE/1000 session. It may also be helpful in diagnosing startup problems. The author hopes that the information in the article will help you understand RJE/1000 better and use it more effectively.

RJE/1000 Initiation Flowchart


RU,RJE,0,0,101
schedule RJE to receive configuration records from the system console
RJE returns its # prompt
Enter #S,T,ON
Switch on TRACE and/or diagnostic messages mode
RJE returns #
Enter 1,1,1
Set input, list, and punch data streams to all be at the system console
RJE: 10 ON
RJE status message
RJE: DIAL REMOTE
Message from #DIAL program
Press Talk Button
DIAL REMOTE PORT
BUSY?
YES
NO
Phone Answered, Now Hear Tone?
NO
YES
A
Call DP Center — their Communications Controller is down.
TRY Again?
YES
NO
OF,RJE,1


A
Press Data Button — It should remain lit from now on.
RJE returns ●
Enter /*SIGNON REMOTE71 12345678
Enter Sign-on Card
Does ● prompt appear within two minutes?
NO
YES
RJE:41 Remote does not respond to bid for line
RJE:72 Terminated
Call DP center, IBM host is down
Enter ●P command
This command sends the sign-on card
When RJE returns ●, is Data Button still lit?
NO
YES
SUCCESS! Connection has been established
Call DP center, IBM host rejected the sign-on card
Enter a garbage record so that RJE can detect the downed line.
RJE:34 Loss of Data Set Ready signal
RJE:72 Terminated

Note:

This flowchart should help you distinguish between the 3 typical failure modes that would result in the need to call the DP Center.

MAINTAINING A VALID DATA BASE

Erik M. Best
General Electric Broadcasting
1400 Balltown Road
Schenectady, New York 12309

After reading the article on "Debugging IMAGE" in November's issue of the COMMUNICATOR, it occurred to the author that there is another important aspect to the problem of keeping your IMAGE system up and running. First, as many program bugs and application misuses as possible must be eliminated. Mr. Field's suggestions should prove to be excellent guidelines with which to attack this area. The second aspect all IMAGE users must face is that of rectifying a corrupt data base when it does occur.

After a System Crash

For example, sooner or later we will all experience the frustration of a system crash just at the end of a long day's processing. The integrity of the data base is obviously lost when the only way out of the situation is to re-boot. After a short investigation it may appear that a well exercised production program has gone awry. Chances are that the data base is corrupt and the damage is compounding itself with every additional transaction. Now you are faced with the question of what to do about it. If you are lucky, your operation can survive the rest of the day with a less than up-to-date data base. Later you need only spend the time required to reload a valid data base, and re-enter or by some other means recover the lost data. But, if your application is one such as ours, the day does not end without the successful completion of all of the day's transactions.

One lost day is bad enough. Unfortunately, it is very possible that a flaw sustained by the data base will not become apparent until days later. In such a case your recovery problem of one day is now compounded by several days. When these situations occur you can spend some long days, nights, or weekends recovering your data base.

It has been our experience that if you can detect a problem early, before it has had a chance to spread, you not only minimize the time you may spend operating with a bad data base, but also shorten your recovery time. In fact, you might even be able to save it. In cases of minimal damage you could very well save yourself hours of recovery time by simply repairing the data base in place. This procedure demands that one be more than familiar with the chaining structure and data set formats of the IMAGE system. Fortunately, armed with an IMAGE manual and a few articles from past issues of the Communicator/1000, it is not such a difficult and mysterious task as you might think.

Detecting a Corrupt Data Base

At this point the author suggests that a few hours spent writing a non-IMAGE validation routine will serve a twofold purpose. First, it will force you to become more familiar with the IMAGE structure previously mentioned. Secondly, when you are done you will have the all-important tool necessary to detect a corrupt data base as early as possible.

One might wonder why anyone would bother to write a validation program when DBSPA could be used. There are several good reasons. While DBSPA will detect any discrepancies between the number of records actually used and what the root file "thinks" is used, there are several other types of problems which would go undetected. In addition, DBSPA leaves you with nothing but the set name from which to work. Your validation program would report exactly where it has encountered the problem, and the nature of it. The run time of such a routine could be comparable with that of DBSPA.

To do the job you must be familiar with Type 2 files, FMGR file handling, IMAGE structure. This author suggests you include the features described below.

> Start with a serial read of each master, following its detail chains to their completion while comparing each key item to the master.

End your check on each master entry with a comparison of your findings with the detail count and last record address for that chain held in the master.

Serially read each detail set, checking constantly that all addresses in the used records are within the limits of the set.

Ascertain the validity of the next available address found in the open records.

When you are finished with all the detail sets, cross check total counts of used records found in both the master and detail reads.

End with a comparison of these totals to the differences between capacity and free record counts in the root file.

With such a program you will have a diagnostic that will save you multifold the time you spent preparing it. You will not have to search your data base for the problem areas and you will be confident that the data base is valid. You now have an early warning system that will alert you if a problem begins to develop. Do not run your validation when the data base is open; the result may be devastating. False errors will show up because the data sets will not reflect their true current status due to unposted IMAGE DCB's. You should find a time of day to run it that suits your particular operation best. Perhaps each night after closing the data base would be a good time.

So you discover that you have a bad data base. What then? The next step is to investigate the bad chain or records to the fullest extent. To help you in this venture the author suggests one more relatively simple FMGR program. A routine should be written to enable you to list records from any data set in an interactive mode. Be sure that it separates the media portion of the records from the data entry. Starting with your validation results a short session spent in investigation should give you a good grasp on the extent of the problem. If the damage is slight, i.e. a broken chain or neglected master address update, it is well worth trying to fix the data base by patching with your own utility or perhaps CMM4. Now re-run your validation, or test the chain via some IMAGE exercise program and you may find that you have saved yourself hours of reprocessing.

Reconstructing the Data Base

Unfortunately, in a large percentage of cases you may find that your data base is in critical condition. Here is where the fastest possible recovery procedure is essential. A quick mention of RECOV: if your only fault was failure to close the data base, either through neglect or an abort, and if you had the data base open in Mode 2, then perhaps this utility will save you. However, these circumstances may not always be the case. It is also true that a DBULD/DBLOD of the entire data base can resolve your problem with a minimal loss of data. Unfortunately, short of a full week's worth of reprocessing, it is by far the most time consuming alternative. Some other method of recovery might be desired.

Ideally one of the fastest reconstruction methods would be to reload a valid data base and simply duplicate all the IMAGE DBPUT's, DBUPD's, and DBDEL's that occurred since your last dump. However, complications arise when you consider that after a system crash logical transactions from your operation which constitute several IMAGE calls may be only partially completed. Only one part of one transaction's IMAGE calls could be duplicated in recovery, still leaving you with a discrepancy in your data.

A second solution is relatively slower than direct IMAGE calls, but is better than manual re-entry. It requires that your application system be tailored with recovery in mind. When the time comes within each program to perform an IMAGE update you would then pass this data to a separate segment. This segment's only purpose would be to set up the necessary IMAGE calls required by the particular update transaction. At the same time you must record this data (transaction) on some storage medium such as tape, cassette, or disc. Whenever recovery becomes necessary you reload a valid data base and feed the recorded transactions back into what essentially would be the identical segments which originally made each update.

Using this technique you can avoid any unnecessary user or data acquisition logic. Your transactions re-occur in the same order as originally run, and since each logical transaction is duplicated in its entirety you avoid any partially-updated entry problems. There are still other techniques that might be used to recover an on-line system such as disc snapshots or frequent disc dumps, however, each may present unapproachable problems in any particular operation.

Fast Recovery for Large Data Bases

You noticed that recovery via the transaction reprocess method starts with the reload of a valid data base. Anyone working with a data base of any substantial size, e.g. 50,000 records or more, knows that a DBRST (and DBSTR for that matter) can take a significant amount of time, and the time factor is critical when users are waiting. What might be needed is a quick and efficient method of dumping and restoring the files. This author successfully adopted such a procedure, which will be described below.

Basically it requires that you establish your data base on a clean cartridge so that your first set will begin on track 0, sector 0. This positioning will eliminate any future movement caused by packing the cartridge. Then calculate the number of blocks remaining after the last data set to the end of that particular track. Create a dummy fill file, Type 1, with the same security code as your data base and just large enough to fill the rest of that track. You then require just two simple programs, one to read the required tracks to tape (or another disc) and another to write them back to the original position on the disc, each one reading and writing a full track at a time. You will find that you can now store and restore your data base in just a fraction of the time that it previously required.

You may want to rewind and reread each dump tape for verification, and if you subsequently change the size of your data base you must adjust the track numbers in your dump and load programs. A final noteworthy advantage to this technique is that you can write a header record on your dump tape which may include the data base name and security code along with the date and time of day. Then read and display this information with a go/no-go option when doing a reload, thus ensuring your knowledge of what is being reloaded.

Hopefully your data base will always remain healthy, but problems do arise. If you use a little initiative and imagination you will have a good headstart towards re-starting your operation in the shortest possible time.

CSD'S CUSTOMER HARDWARE MAINTENANCE TRAINING HP 1000 SERIES COMPUTER/HP DISC DRIVE

CSD's Customer Maintenance Training is conducting two newly structured Maintenance programs dealing with the HP 1000 computer family and HP Disc Drives.

The HP 1000 Operation and Maintenance course (91303A) is an 8-day program covering operation and maintenance of all members of the 1000 computer family. The 8-day format is unique in that each class starts on a Tuesday and ends on Thursday of the following week. This allows for travel on the Monday preceeding the class, and on Friday following the class.

The HP Disc Drive Operation and Maintenance course (91304A) is an 8-day program covering operation and maintenance of either the 7905/6 or 7920/5. 7905/6 training is specified by ordering Option 001, 7920/5 training is Option 002. This course is structured with the first two days being strictly lecture. The remaining three days will be in a lab environment wherein the students concentrate on their selected disc drive. Sound-On-Slide maintenance programs are available as training aids during this period. Any students desiring training on both 7905/6 and 7920/5 must attend two classes. However, the student need only attend the lab portion (last three days) of the second class.

For registration information, please contact your local sales office.

HP COMPUTER SERVICE DIVISION TRAINING

91304A

HP DISC DRIVE MAINTENANCE
7905/6 OPTION 001
7920/5 OPTION 002

PREREQUISITE: Digital electronics background and prior lecture segment if only lab segment is taken.

PURPOSE: To provide the student with the fundamental knowledge needed to troubleshoot and repair to the replaceable sub-assembly level, adjust, align, and maintain the HP Multi-Access Controller Disc Drive family using the HP 13354-60005 Disc Service Unit and diagnostics.

LENGTH: 2 days Lecture and either option — 3 days Lab, 7905/6 (option 001) — or 3 days Lab, 7920/5 (option 002)

LECTURE: 2 days general lecture on the HP Multi-Access Controller Disc Drive family and subsystem.

LAB: 3 days lab time provides each student with the opportunity for hands-on experience with alignment and preventive maintenance procedures, troubleshooting to the replaceable sub-assembly level and replacement of faulty sub-assemblies.

COURSE CONTENT:

Theory of operation.

Alignment procedures.

Preventive maintenance.

Replacement procedures.

Use of the Disc Drive Service Unit for alignment and preventive maintenance.

Use of diagnostics.

Troubleshooting techniques.

HP COMPUTER SERVICE DIVISION TRAINING

91303A

HP 1000 SERIES COMPUTER MAINTENANCE

PREREQUISITE: A strong background in digital electronics, including binary, octal, and decimal number system conversions. Knowledge of machine/assembly language programming is also desirable.

PURPOSE: To provide the student with specific theory and lab exposure needed to troubleshoot, repair by major subassembly replacement, and maintain the HP 1000 M/E/F Series Computers. The course covers an introduction to computer programming in machine language; the implementation of the standard HP 1000 Series instructions; the hardware of the HP 1000 Series computer; and operation of the 2748B paper tape reader, 2895 paper tape punch, and 2645A CRT Terminal.

LENGTH: 8 days (starting on Tuesday and ending on Thursday).

LAB: Lab time provides each student with the opportunity for hands-on troubleshooting and subassembly replacement experience.

COURSE CONTENT:

Front panel operation.

Machine language programming.

Tape reader, punch and terminal operation.

Diagnostic operation.

Initial binary loader.

Preventive maintenance.

I/O structure.

Power supply adjustment and replacement.

Block diagrams.

Memory.

Dual channel port controller.

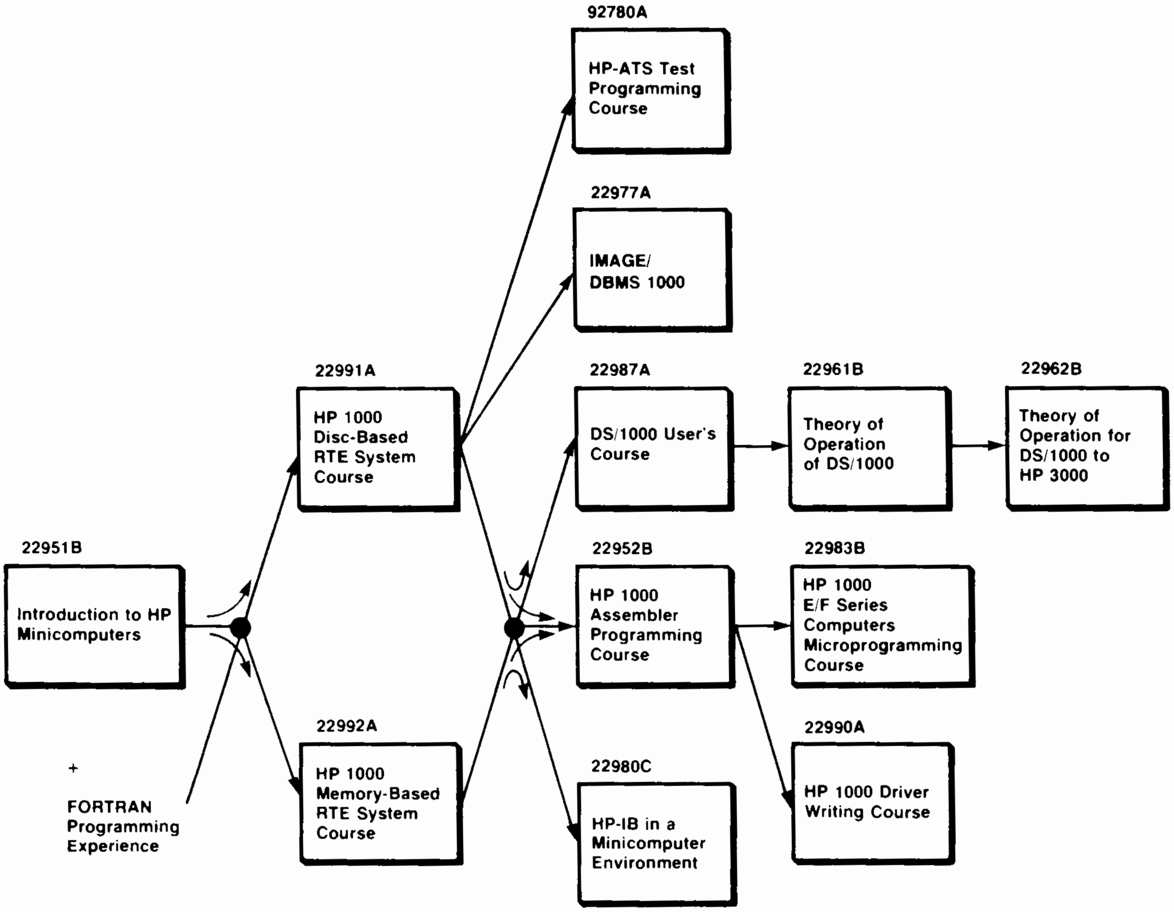
Memory protect.

Dynamic Mapping system.

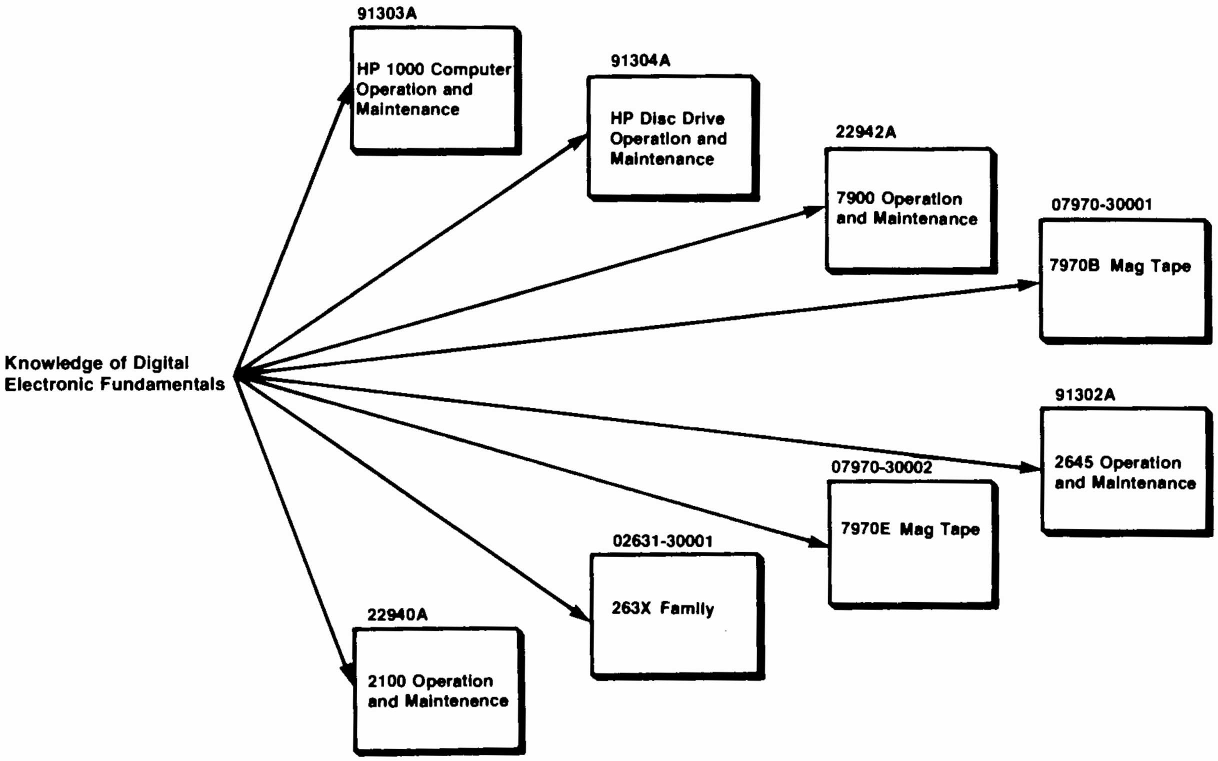
TECHNICAL SYSTEMS COURSE PREREQUISITES

(Prerequisites must be satisfied in a left to right order OR equivalent experience)

HP 1000 PROGRAMMING COURSES


92780A
HP-ATS Test Programming Course
22977A
IMAGE/ DBMS 1000
22991A
HP 1000 Disc-Based RTE System Course
22987A
DS/1000 User's Course
22961B
Theory of Operation of DS/1000
22962B
Theory of Operation for DS/1000 to HP 3000
22951B
Introduction to HP Minicomputers
+
FORTRAN Programming Experience
22952B
HP 1000 Assembler Programming Course
22983B
HP 1000 E/F Series Computers Microprogramming Course
22992A
HP 1000 Memory-Based RTE System Course
22980C
HP-IB in a Minicomputer Environment
22990A
HP 1000 Driver Writing Course

HP 1000 MAINTENANCE COURSES


91303A
HP 1000 Computer Operation and Maintenance
91304A
HP Disc Drive Operation and Maintenance
22942A
7900 Operation and Maintenance
07970-30001
7970B Mag Tape
Knowledge of Digital Electronic Fundamentals
91302A
2645 Operation and Maintenance
07970-30002
7970E Mag Tape
02631-30001
263X Family
22940A
2100 Operation and Maintenence

COURSE INDEX

GENERAL COURSES

		ATLANTA	BOISE	CHICAGO	DALLAS	DETROIT	LOS ANGELES	MONTREAL	NEW YORK	SAN FRANCISCO Comp. Serv. Div	SAN FRANCISCO Sys. Eng. Div.	SAN FRANCISCO Data Sys. Div.	SEATTLE	TORONTO	WASHINGTON, D.C.
22967B	Introduction to Programming				●										

TECHNICAL COURSES

HP 1000 Courses

		ATLANTA	BOISE	CHICAGO	DALLAS	DETROIT	LOS ANGELES	MONTREAL	NEW YORK	SAN FRANCISCO Comp. Serv. Div	SAN FRANCISCO Sys. Eng. Div.	SAN FRANCISCO Data Sys. Div.	SEATTLE	TORONTO	WASHINGTON, D.C.
22951B	Introduction to HP 1000 Minicomputers			●			●	●	●		●			●	●
22991A	Disc-based RTE System Course	●		●	●	●	●	●	●		●			●	●
22992A	Memory-based RTE System Course										●				
22977A	IMAGE/DBMS 1000			●			●		●		●				●
22952B	HP 1000 Assembler Programming			●			●		●		●				●
22987A	DS/1000 User										●				●
22961B	DS/1000 Theory of Operation														●
22962B	DS/1000 to HP 3000 Theory of Operation														●
22980C	HP-IB in a Minicomputer Environment										●				
22983B	E & F Series Microprogramming								●		●				
22990A	RTE Driver Writing			●					●		●				●
22965B	RTE II/III Operating System														●
RTE-U	RTE-IV Upgrade			●											
92780A	HP-ATS Automatic Test System											●			
	Advanced RTE Workshop										●				●
T2647	HP 2647 Terminal Course						●								

MAINTENANCE COURSES

		ATLANTA	BOISE	CHICAGO	DALLAS	DETROIT	LOS ANGELES	MONTREAL	NEW YORK	SAN FRANCISCO Comp. Serv. Div	SAN FRANCISCO Sys. Eng. Div.	SAN FRANCISCO Data Sys. Div.	SEATTLE	TORONTO	WASHINGTON, D.C.
02631-30001	263X Family Course		●												
07970-30001	7970B Mag Tape Course		●												
07970-30002	7970E Mag Tape Course		●												
22940A	2100 Operation & Maintenance									●					
22942A	7900 Operation & Maintenance									●					
91302A	2645 Operation & Maintenance									●					
91303A	HP 1000 Operation & Maintenance									●					
91304A	HP Disc Drive Operation & Maintenance									●					

TRAINING SCHEDULE

The current schedule for customer training courses on HP 1000 computer systems products is given in this section. Included are courses offered in the U.S., Europe, and in International areas during the upcoming months.

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 U.S. training centers only. For prices of courses at European or International training centers please consult your local HP sales office.

DATA SHEETS

Data sheets giving detailed information on each of the courses scheduled are available from your local HP representative.

REGISTRATION

To enroll in any of the courses listed in this publication please contact your local HP sales office and provide them with a class name, number and date you wish to attend along with a purchase order number from your company.

ADVANCED REGISTRATION

Hewlett-Packard training centers accept advanced registration for all courses. However, if a purchase order from your company has not been received at least two weeks prior to the start date of your class, reservation cannot be guaranteed.

ACKNOWLEDGEMENT

Within 10 working days of the receipt of your registration request, you will be sent a letter of confirmation and other local area information to help you plan your own hotel and travel accommodations.

CANCELLATION

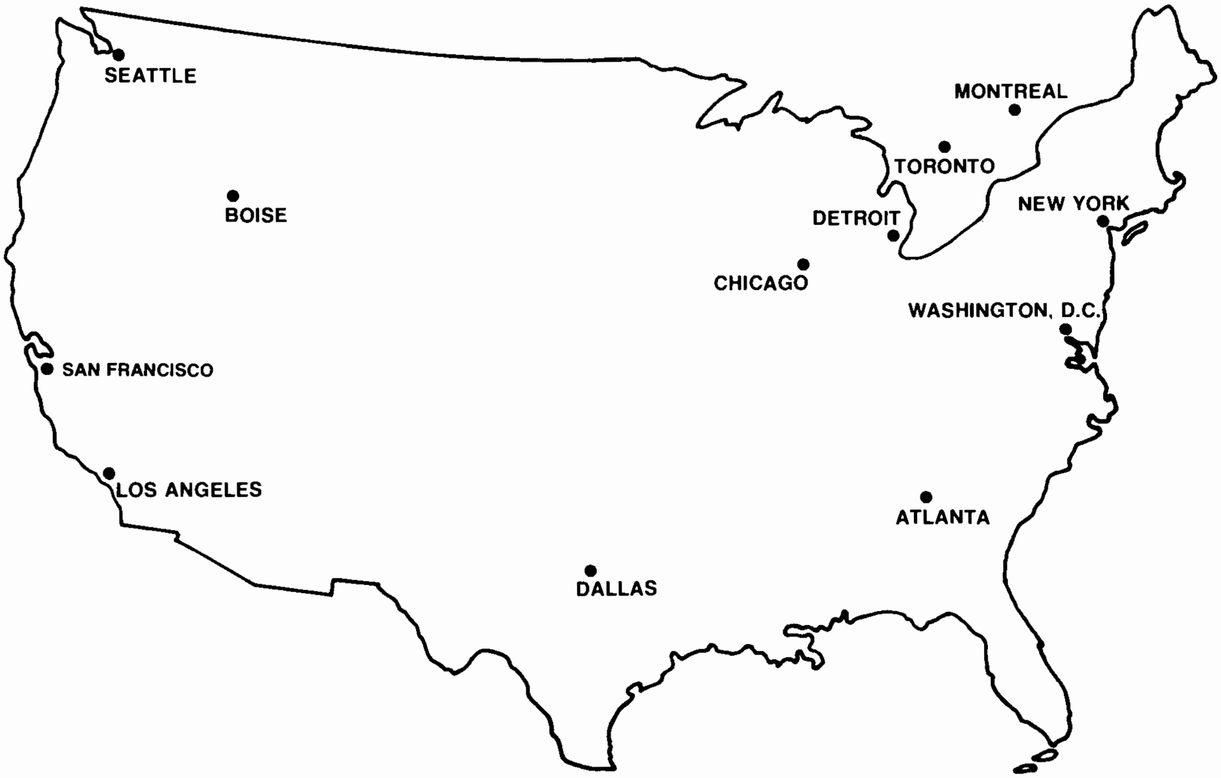
Hewlett-Packard reserves the right to cancel any class due to insufficient enrollment. If this should occur, all enrollees will be notified as soon as possible in order to make other plans.

NORTH AMERICAN TRAINING CENTER LOCATIONS

The location of each North American training center is shown on the following page. Detailed addresses and phone numbers are given along with the schedule for each center.

Computer Systems
North American
Customer Training Schedules

Spring-Summer 1979



U. S. TRAINING CENTER SCHEDULES, LOCATIONS, AND RATES

Course Number	Title	Length	Price	SAN FRANCISCO Systems Engineering Center	LOS ANGELES Systems Engineering Center	WASHINGTON D. C. Systems Engineering Center	CHICAGO Systems Engineering Center	DALLAS (D) ATLANTA (A) Systems Engineering Centers	DETROIT Systems Engineering Center	NEW YORK Systems Engineering Center
22951B	Intro to HP mini's	4 days	400	Apr 2	Mar 12 Apr 30	Mar 19 Apr 16	Apr 2	May 21 (D)	Mar 26 May 14	Mar 5 May 7
22991A*	HP 1000 DISC RTE (Course includes RTE-IV operating system, batch spool monitor and file manager.)	10 days	1000	Mar 12 Mar 26 Apr 16 Apr 30 May 14 Jun 4	Mar 19 Apr 16 Mar 7	Mar 12 Mar 26 Apr 16 Apr 30 May 14	Apr 16	Jan 29 (A) Mar 19 (D)		Mar 12 Apr 16 May 14
22992A*	HP 1000 Memory RTE	10 days	1000							
22977A*	IMAGE	5 days	500	Apr 23	May 21	Mar 12 May 21	Apr 30			Apr 2
22952B*	1000 ASMB	5 days	500	Apr 30	Jan 29 Apr 2 Jun 4	Jan 29 Mar 5 Apr 23	Mar 5			Mar 26 Apr 30
22987A*	DS/1000 User's Course	5 days	500	Mar 26		Jan 8 Apr 30				
22961B*	DS/1000 Theory of Op.	4 days	400			Jan 15 May 7				
22962B*	DS/1000 to HP 3000 Theory of Op.	1 day	100			Jan 19 May 11				
22990A*	RTE-Driver Writing	3 days	300	May 21		May 30				Apr 9

*These courses carry prerequisites — refer to the training program diagram and discussion on the previous pages for more information.

U. S. TRAINING CENTER SCHEDULES, LOCATIONS, AND RATES (Continued)

Course Number	Title			SAN FRANCISCO Systems Engineering Center	LOS ANGELES Systems Engineering Center	WASHINGTON D. C. Systems Engineering Center	CHICAGO Systems Engineering Center	DALLAS (D) ATLANTA (A) Systems Engineering Centers	DETROIT Systems Engineering Center	NEW YORK Systems Engineering Center
		Length	Price							
22980C*	HP-IB Minicomputer Environment	4 days	400	Mar 19						
22983B*	HP 1000 E/F Microprogramming	5 days	500	May 7						May 14
	Advanced RTE Workshop	5 days	800	Mar 12 May 14						

*These courses carry prerequisites — refer to the training program diagram and discussion on the previous pages for more information.

**For Registration Information, call Cupertino Customer Training Center Registrar.

U. S. TRAINING CENTER SCHEDULES, LOCATIONS, AND RATES (Continued)

Course Number	Title		Data Systems Division (SAN FRANCISCO)	Data Terminals Division (CUPERTINO)	Customer Service Division (CUPERTINO)	Boise Division (BOISE)	
	Length	Price					
92780A*	HP-AS Automatic Test System						
	5 days	1000					
13294A	Dev. Terminal						
	5 days	500					
22940A	2100 Maint.				Mar 26 Apr 30		
	10 days	1000					
91303A	HP 1000 Operational Maintenance				Mar 6 Mar 27 Apr 17 May 15		
	8 days	1000					
22942A	7900 Maint.				Apr 23		
	5 days	500					
91304A	HP Disc Drive Operational Maint.				Mar 12 Mar 19 Apr 16 May 14 May 21		
	5 days	500					
91302A	2645 Maint.						
	3 days	300					
22943A	7970B Maint.						
	5 days	600					
22944A	7970E Maint.						
	5 days	600					

*These courses carry prerequisites — refer to the training program diagram and discussion on the previous pages for more information.

U.S. TRAINING CENTER ADDRESSES

Atlanta
CUSTOMER TRAINING CENTER
450 Interstate North Parkway, NW
Atlanta, Georgia 30339
(404) 955-1500

Boise
BOISE DIVISION
11311 Chinden Boulevard
Boise, Idaho 83702
(208) 377-3000

Dallas
CUSTOMER TRAINING CENTER
201 E. Arapaho Road
Richardson, Texas 75080
(214) 231-6101

Los Angeles
CUSTOMER TRAINING CENTER
1430 E. Orangethorpe Avenue
Fullerton, CA 92631
(714) 870-1000

New York
CUSTOMER TRAINING CENTER
120 Century Road
Paramus, N.J. 07652
(201) 265-5000

San Francisco
CUSTOMER TRAINING CENTER
19320 Pruneridge Avenue
Cupertino, CA 95014
(408) 996-9800

DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, CA 95014
(408) 257-7000

DATA TERMINALS DIVISION
19400 Homestead Road
Cupertino, CA 95014
(408) 257-7000

CUSTOMER SERVICE DIVISION
19310 Pruneridge Avenue
Cupertino, CA 95014
(408) 996-9383

Washington, D.C.
CUSTOMER TRAINING CENTER
4 Choke Cherry Road
Rockville, MD 20850
(301) 948-6370

EUROPEAN TRAINING CENTER SCHEDULES AND LOCATIONS

Course Number	Title Length	Böblingen	Amsterdam	Madrid	England Altrincham (A) Winnersh (W)	Milan (M) Rome (R)	Stockholm	Helsinki	Orsay	Vienna	Brussels
22951B	Intro to HP mini's 4 days 400	Mar 05 May 07 Jul 16 Sep 24	Apr 30 Aug 27		Feb 26 (W) Mar 19 (A)		Apr 02 Oct 08			Sep 03	
22965B	RTE-II/III 10 days (Course includes RTE-II/III operating system, batch spool monitor and file manager.)	Mar 12 Jun 18							Mar 26 May 07 Jun 11		
22991A*	HP 1000 DISC RTE 10 days 1000 (Course includes RTE-IV operating system, batch spool monitor and file manager.)	Feb 19 Mar 26 Apr 23 May 07 May 21 Jul 09 Jul 30 Aug 20 Sep 10 Oct 01 Oct 15	Feb 26 Apr 02 May 14 Jun 25 Sep 03 Oct 15	Feb 26 Oct 29	Feb 05 (A) Mar 04 (W) Mar 26 (A)		Mar 05 Apr 23 Sep 10 Oct 15 Nov 19	Mar 05 Apr 30		Mar 19 Sep 10	Mar 12 May 28 Oct 01
22985A	RTE-M 5 days	Jun 18 Sep 10							Mar 5		
22977A*	IMAGE 5 days	Mar 26 Jul 09	Apr 23 Jul 16 Oct 08	Mar 12 Nov 12	Apr 23 (A)			May 21	Mar 12 Jun 25 Sep 24	Apr 02	
22952B*	1000 ASMB 5 days	Apr 02 Jun 25 Sep 03	Mar 19 Jun 11 Oct 01	Feb 19 Oct 22	Feb 19 (A)		Feb 26 Sep 03 Nov 12	Feb 19	Mar 05 May 08	Sep 24	
22987A*	DS/1000 User's Course 5 days	May 28 Sep 17	Sep 17		Mar 26 (W)						
22961B*	DS/1000 Theory of Op. 4 days	Mar 12 Oct 01	Sep 24		Apr 02 (A)						
22962B*	DS/1000 to HP 3000 Theory of Op. 1 day	Mar 16 Oct 05	Sep 28		Apr 06 (A)						
22990A*	RTE Driver Writing 3 days										
22980C*	HP-IB Minicomputer Environment 4 days	Jul 23	Apr 16 Aug 20						Apr 17		

*These courses carry prerequisites — refer to the training program diagram and discussion on the previous pages for more information.

EUROPEAN TRAINING CENTER SCHEDULES AND LOCATIONS (Continued)

Course Number	Title Length	Böblingen	Amsterdam	Madrid	Winnersh	Milan (M) Rome (R)	Stockholm	Grenoble	Orsay	Vienna	Brussels
22983B*	HP 1000 E/F Microprogramming 5 days										
92780A*	HP-ATS Automatic Test System 5 days										
13294A	Dev. Terminal 5 days										
22940A	2100 Maint. 10 days										
22941A	21MX/XE Maint. 5 days							Feb 26 Jul 02 Oct 22			
22942A	7900 Maint. 5 days							Mar 5			
22945A	7905/06 Maint. 5 days							Feb 12 Apr 30 Jul 09 Oct 29			
22984A	7920 Maint. 5 days							Feb 19 May 07 Jul 16			
91302A	2645 Maint. 3 days							May 28			
22943A	7970B/E Maint. 5 days							Mar 26 Jul 23			
40270A	Intro to HP Computers 5 days							Apr 09 Jul 02			
22965B-H01	FORTRAN IV 5 days	Mar 19 May 14 Jul 23 Oct 08		Oct 15							

*These courses carry prerequisites. Refer to the training program diagram and discussion on the previous pages for more information.

EUROPEAN TRAINING CENTER ADDRESSES

Altrincham, England
Navigation Road
Altrincham
Cheshire WA14 1NU

Amsterdam, the Netherlands
Van Heuven Goedhartlaan 121
Amstelveen 1134
Netherlands
Tel: 02 672 22 40

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

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

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

Helsinki, Finland
Nahkahousuntie 5
00211 Helsinki 21
Tel: 90-692 30 31

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

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

Orsay, France
Quartier de Courtaboeuf
Boite Postale No. 6
F-91401-Orsay
France
Tel: (01) 907 7825

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

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

Winnersh, England
King Street Lane
Winnersh, Workingham
Berkshire RG11 5 AR
Tel: Workingham 784774
Cable: Hewpie London
Telex: 8471789

INTERCONTINENTAL TRAINING CENTER SCHEDULES AND LOCATIONS

Course Number	Title Length	CANADA Montreal	CANADA Toronto	AUSTRALIA Blackburn, VIC (B) Pymble, NSW (P)	JAPAN Tokyo (T) Osaka (O)	
22951B	Intro to HP mini's 4 days	Feb 19**	Jan 08		Feb 27 (T) Apr 10 (T) May 22 (T)	
22991A*	HP 1000 DISC RTE 10 days (Course includes RTE-IV operating system, batch spool monitor and file manager.)	Mar 19**	Feb 19	Feb 26 (B) May 21 (P) Jul 09 (B) Sep 17 (P) Oct 22 (B)	Mar 05 (T,O) Apr 16 (T,O) May 28 (T) Jun 04 (O)	
22992A*	HP 1000 Memory RTE 10 days				Feb 19 (T)	
22977A*	IMAGE 5 days			Apr 30 (B) Jul 09 (P) Aug 20 (B) Nov 12 (P) Dec 03 (B)	Jun 25	
22952B*	1000 ASMB 5 days			Jun 11 (P) Jul 30 (B) Oct 08 (P) Nov 12 (B)	Mar 26 (T) Apr 16 (O) May 07 (T) Jun 11 (T)	
22987A*	DS/1000 User's Course 5 days			May 28 (B) Oct 22 (P)	Jun 18 (T)	
22961B*	DS/1000 Theory of Op. 4 days					
22962B*	DS/1000 to HP 3000 Theory of Op. 1 day					
22990A*	RTE-Driver Writing 3 days			Jun 18 (P) Aug 06 (B) Oct 01 (P) Nov 19 (B)	Apr 03 (T)	
22980C	HP-IB 4 days				May 14 (T)	

*These courses carry prerequisites — refer to the training program diagram and discussion on the previous pages for more information.

**These courses are taught in French.

INTERCONTINENTAL TRAINING CENTER ADDRESSES

Blackburn, Australia
CUSTOMER TRAINING CENTER
31-41 Joseph Street
Blackburn, Victoria, Australia

Pymble, Australia
CUSTOMER TRAINING CENTER
31 Bridge Street
Pymble, New South Wales, Australia

Montreal, Canada
CUSTOMER TRAINING CENTER
275 Hymus Boulevard
Pointe Claire, Quebec, Canada H9R1G7
(514) 697-4232

Toronto, Canada
CUSTOMER TRAINING CENTER
6877 Goreway Drive
Mississauga, Ontario, Canada, L4V 1M8
(416) 678-9430

NEW PRODUCT ANNOUNCEMENTS

ANNOUNCING DOUBLE PRECISION (64-BIT) FLOATING POINT CAPABILITY FOR F-SERIES COMPUTER, FORTRAN

Bill Elmore/HP Data Systems Division

An important enhancement to the HP 1000 F-Series computer, double precision (64-bit) hardware floating point operations, is fully supported as of January 1, 1979. This new capability is the result of enhancements to FORTRAN and the RTE library, which are summarized below.

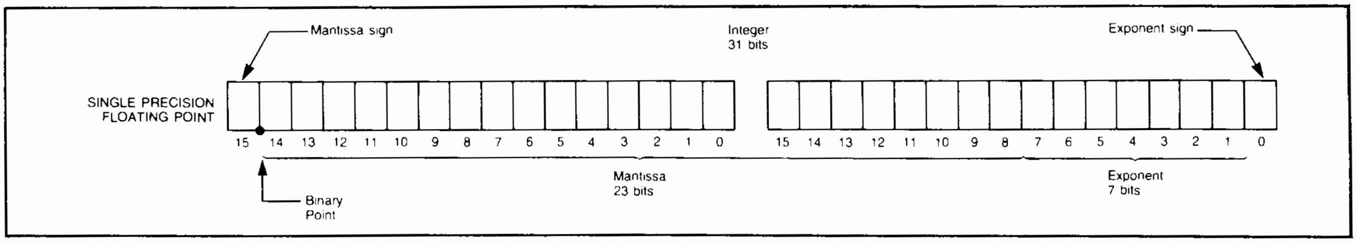
- Double precision (64-bit) instructions for addition, subtraction, multiplication, division, and all floating point-integer conversions. This provides 17 significant decimal digits of precision.
- Execution at hardware speed in F-Series floating point processor.
- Full FORTRAN support of all double precision operations.
- FORTRAN enhancements allow double precision execution at the same speed at today's extended precision (48-bit) operations.
- Double precision versions of all mathematical functions have been added to the RTE Library.
- Software equivalent arithmetic functions allow double precision operations on HP-1000 E-Series and M-Series computers.

and best of all

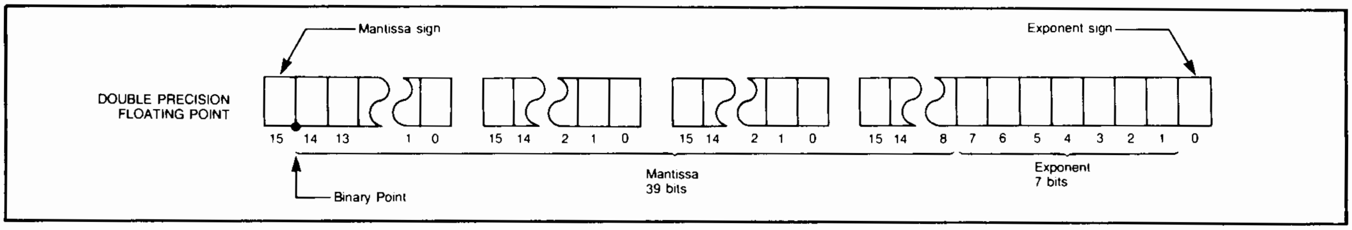
- No hardware changes required for existing F-Series computers!

What is Double Precision?

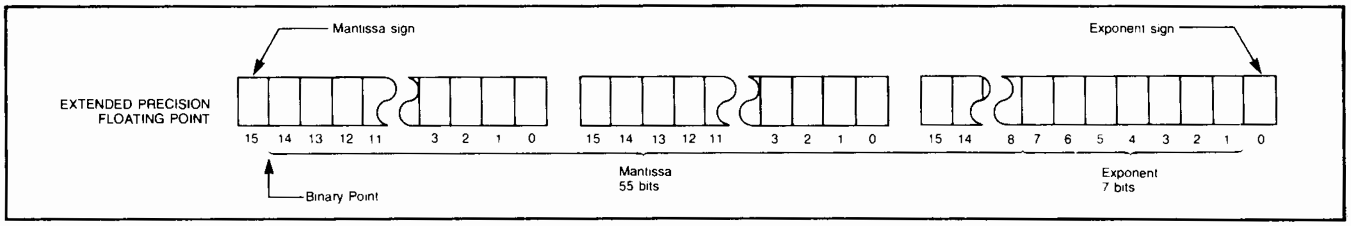
Floating point numbers are represented inside the computer as a fractional part (the mantissa) times an exponent. The precision of the floating point number is determined by the length of the mantissa. For example, a single precision floating point number consists of two 16-bit words of which 23 bits represent the mantissa. This provides approximately 7 significant decimal places of precision.



If more than seven significant digits are required, extended precision format is available. An extended precision floating point number consists of three 16-bit words and uses a 39-bit mantissa for 11 significant decimal digits of precision.



And now comes double precision which uses four 16-bit words to represent a floating point number. Double precision has a 54-bit mantissa with a precision of 17 decimal digits. Double precision floating point operations are also available on HP 1000 M-Series and E-Series computers through the use of software equivalents of the new F-Series instructions.



The HP 1000 Floating Point capabilities are summarized below.

	Single Precision	Extended Precision	Double Precision
F-Series	Hardware	Hardware	Hardware
E-Series	Firmware	Firmware (optional)	Software
M-Series	Firmware	Firmware (optional)	Software

Ordering Information

Customers who now have an HP 1000 F-Series Computer should order a new Operating & Reference Manual (Part No. 02111-90001) which explains how to use new double precision instructions. No hardware or firmware changes are required.

Our RTE customers who have purchased Software Subscription Service or Comprehensive Software Support will receive an enhanced FORTRAN Compiler and RTE Library routines that provide double precision floating point capabilities along with the corresponding software manual updates.

Customers without Software Subscription Service or Comprehensive Software Support should order 92067A-001 to get a totally up-to-date version RTE-IV operating system.

INTRODUCING THE NEW 91731A ASYNCHRONOUS MULTIPLEXER SOFTWARE

Tony Russo/HP Data Systems Division

The 91731A is a software support package for multiplexed terminal communication between an HP 1000 Computer System operating under RTE-MIII or RTE-IV and Hewlett-Packard CRT Display and printing terminals, using the 12920B 16-channel Asynchronous Multiplexer interface. At $2000, U.S. List Price, the 12920B 16-channel multiplexer is an extremely cost effective way to support many widely scattered HP terminals at low speeds over asynchronous communication lines.

Some of the features of the new 91731A software include:

- Supports up to two 12920B multiplexer interfaces, up to 16 channels per 12920B
- Program development and/or application program execution on RTE-IV systems
- Application program execution on RTE-MIII systems
- Support for 2621A/P, 2631A, 2635A, 2640B, 2645A, and 2648A terminals and Bell type 103A2 and 212A Data Sets and Vadic VA3400 modems
- Block mode data transfer capability
- Hardwired or full-duplex modem communication support
- Support for 2645A/48A Mini cartridge tape units and/or auxiliary printers
- Individually programmable channel characteristics
- Split speed operation with 264xA/B terminals
- Defined subsystem performance characteristics

Capabilities **not** supported by the 91731A are:

- The system console
- Current loop operation
- Half duplex operation
- User-written logical drivers
- Parity on binary data when using Mini cartridges
- Power fail recovery after power fail
- Dialing on 801 Auto Calling Unit (although 12920B interface is capable of that)
- Plotting or reverse line feed on 13349A auxiliary line printer for 2645A/2648A terminal

Maximum throughput for the 91731A/12920B Multiplexer Subsystem is as follows (all figures are aggregate throughput):

M-Series	590 characters/second
E-Series	1000 characters/second
E/F-Series with HS Memory	1200 characters/second

How to Order the 91731A

The 91731A software subsystem is priced at $250 U.S. list price and comes only on mini-cartridge (option 020). The 91731A + 020 package includes:

1. 91731A software on mini-cartridge
2. 91731A User's Guide (91731-90001)
3. 91731A Software Numbering Catalog (91731-90002)
4. 91731A Configuration Guide (91731-90003)

Also available is the Software Subscription Service (91731S + 020) at $10/month and the Comprehensive Software Support (91731T + 020) at $20/month. Both of these products are supplied on mini-cartridge only.

Note that the 91731A is a Type II software product; a customer may copy the software once for each HP 1000 computer purchased.

7225A GRAPHICS PLOTTER NOW SUPPORTED ON GRAPHICS/1000 GRAPHICS PLOTTING SOFTWARE

Mike Scott/HP Data Systems Division

The January 1, 1979 (1901) software release of the 92840A Graphics Plotting Software will add support for San Diego Division's 7225A Graphics Plotter. GRAPHICS/1000 now supports six versatile and cost-effective graphics peripherals:

Software Devices

2648A Graphics Terminal

Hardcopy Devices

7221A Graphics Plotter (RS232, 4-Pen, 11" × 17")
7225A Graphics Plotter (HP-IB with 17601A, 1-Pen, 8-1/2" × 11")
7245A Graphics Plotter/Printer (HP-IB, 8-1/2" Thermal roll paper)
9872A Graphics Plotter (HP-IB, 4-Pen, 11" × 17")

Input Devices

9874A Digitizer

The 7225A "Miniplotter" was recently introduced by the San Diego Division to provide a lower cost solution to the need for professional hard copy graphics. This 18 pound (8 Kg) desktop plotter produces notebook size graphs on 8-1/2" × 11" or ISO A4 sheets. The user can manually change the single pen to produce the same high quality, multicolor plots available from the larger (11" × 17") HP plotters, the 9872A and 7221A.

HP-IB Personality Module — 9872A Compatible

The 7225A has the I/O flexibility to be adopted to different computers and controllers. By changing a user "plug-in" unit, called a Personality Module, the 7225A will provide the appropriate interface, language, and graphics capabilities. With the HP 1000, the 17601A Personality Module provides an HP-IB interface for the 7225A. The 7225A/17601A would then connect to the 59310B HP-IB card in the HP 1000 and use RTE driver DVR37.

The 7225A with the 17601A Personality Module can be run with software written for the 9872A plotter. Since there is only one pen with the 7225A, calls to the GRAPHICS/1000 PEN subroutine will generate different line styles (i.e., Pen 1 = solid line, Pen 2 = light solid line, etc.) unless a call to the LINE subroutine has been made. The HPGL (instruction set understood by the 7225A) command for selecting pen velocity (VS) is ignored since the 7225A has a single plotting speed. The table below summarizes how the 9872A and 7225A compare.

7225A AND 9872A COMPARISON

FEATURE	7225A/17601A	9872A
Paper Size	216 × 279mm (8 1/2 × 11") or ISO A4	280 × 432mm (11 × 17") or ISO A3
Multicolor	Single pen changed manually	Automatic pen selection
Resolution	.032mm (.0013 inch)	.025mm (.001 inch)
Plotting Speed	25 cm/second	User selectable 1 to 36 cm/second
Character Speed	3 cps	3 cps

The 7225A/17601A should really appeal to those customers that have insufficient budgets for the larger 9872A but can afford the 7225A/17601A. Both the 7225A Graphics Plotter and the 17601A Personality Module must be ordered to have a complete plotter.

The 92840A Graphics Plotting Software User's Manual (92840-90001) is being updated for the January, 1979 (1901) PCO so that customers who have SSS (92840S) or CSS (92840T) will receive both the software and manual updates. Customers who already have the 92840A software but not the software support (92840S/T) will have to purchase another copy of the 92840A software if they want 7225A Graphics/1000 support. Software support services are a good investment.

Contact your local HP Sales Office for more information.

NEW EXTENDED PERFORMANCE FOR 2240A

Jim Gruneison/HP Data Systems Division

The HP 2240A Extended Performance Option (Option 001) extends the ability of the Measurement and Control Processor to function independently of the computer that controls it, and to process and return large amounts of data. The features of this option result in a task buffer size effectively three times that of the standard instrument, faster data processing and transfer time, as well as faster processing by the computer. The option provides commands designed to utilize the high-volume high-speed data-handling capability.

Handling of data in binary format increases the speed of data transmission from the 2240A to the computer, saves storage space in the 2240A task buffer, and saves the format processing time associated with ASCII data in both the 2240A and the computer. The Extended Performance commands that utilize these capabilities provide for decision making, nested repeats, and special modes of data acquisition.

The Extended Performance functions make a difference in the way you write program statements to get result data from the HP 2240A. Requests directed to the HP 2240A follow the standard format, but can include operations that are not possible with the standard HP 2240A. This bulletin gives you examples showing how to retrieve result data, and also tells you how to use the additional commands that the Extended Performance Option makes available.

BINARY RESULT DATA

In the standard HP 2240A, the internal formatter converts measurement data to ASCII format, and stores the result in the task buffer. In FORTRAN you use a READ statement to get the data. With Extended Performance, however, no result formatting takes place; the processor stores incoming data directly in the task buffer, and you use a FORTRAN unformatted READ or an EXEC call, to read result data.

For example, a request to measure analog input starting at slot 1, channel 2, and continuing for 4 channels:

```
Request          Result

AI,1,2,4!   0,675,1350,-650,12crlf
```

The result is shown in standard HP 2240A format: condition normal, 675, 1350, −650, and 12 millivolts from channels 2 through 5 respectively. The result items are returned in ASCII format, separated by commas. Each character represents a data byte on the HP-IB. A carriage-return line-feed terminates the data.

The same request made to the HP 2240A with Option 001.

```
Request          Result

AI,1,2,4!     0,675,1350,-650,12
                             EOI
```

The request is identical: commas are optional; spaces or semicolons are acceptable; the "!" is required at the end of the request. The result, however, is quite different. Instead of returning character digits in groups separated by commas, the Extended Performance 2240A sends two binary bytes for each data word, most significant byte first. Each data word is the two's complement binary code for a number between −32768 and 32767. Here are the last two binary data words shown in detail:

```
|<---Binary data word for "-650"--->|<----Binary data word for "12"---->|
|               |               |               |               |
 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0
|               |               |               |               |
|    word 4     |    word 4     |    word 5     |   EOI with    |
|    byte 1     |    byte 2     |    byte 1     | word 5 byte 2 |
```

With the Extended Performance Option, there is no crlf to mark the end of data, and there are no commas between data items. Commas are not needed because each data item is represented by a fixed 16-bit field. Instead of the crlf characters, the 2240A sets End or Identify signal EOI on the HP-IB. This binary data representation is more compact than its ASCII equivalent. This means that more data can be sent from the 2240A-001 to the HP 1000 in any given period of time.

EXTENDED PERFORMANCE COMMANDS

The new commands include a wait command, a command to provide interrupt on error, and a decision-making command pair. Nested repeats are permitted, using the standard **RP . . . NX** command pair to nest repeats up to six deep. An exit command provides conditional branching out of repeat loops. In all, there are five new commands, bringing the number of the high level commands on the 2240A to more than 50. The new commands provide significant programming flexibility for the user.

For Extended Performance data acquisition, a "BIN" mode permits you to accumulate and retrieve data in packages, without terminating the request in process. This BIN mode and binary data transmission results in the following performance capabilities. The table compares the standard 2240A with the new option 001 2240A. As you can notice that the continuous data gathering capabilities have been improved five fold.

Note A **# CHANNELS/ SCAN (N)**	Note B **EXTENDED PERFORMANCE 2240A**		Note C **STANDARD PERFORMANCE 2240A**	
	MAX # SCANS/SEC CONTINUOUS (S1)	**MAX. # RDGS/SEC (N x S1)**	**MAX. # SCANS/SEC CONTINUOUS (S2)**	**MAX. # RDGS/SEC (N x S2)**
16	128 - 240	2048 - 3840	50 - 80	800 - 1280
32	110 - 168	3520 - 5376	35 - 48	1120 - 1536
64	82 - 106	5248 - 6784	21 - 25	1344 - 1600
128	54 - 60	6912 - 7680	12 - 13	1536 - 1664
256	32	8192*	6.5	1664

* Actual Rates with HP 1000 exceed 10000 Rdgs/Sec

Notes:

A. In all cases 256 readings are gathered and stored by the 2240A before shipping to the computer (i.e., bin size in Extended Performance 2240A in 258).

B. Approximate predicted values using following:

1. Approximately 3.5-4.0 ms required every 256 readings for result transmission during which time no readings can be taken.
2. In each case, larger number represents the fastest prediction. Smaller number represents maximum continuous rates using evenly spaced external pulses (sensed by digital input card) to trigger scans. Both take into account all 2240A processing overhead.
3. Maximum average rate at which readings returned to computer main memory.

C. Same as B except 9.0-9.5 ms required every 256 readings for result transmission and re-issue of 2240A request.

HEWLETT-PACKARD
COMPUTER SYSTEMS COMMUNICATOR ORDER FORM

Please Print:

Name ______________________________ Date ______________________

Company ____

Street ____

City __________________________ State ______________ Zip Code ________

Country ____

□ HP Employee   Account Number ____________   Location Code ____________

□ DIRECT SUBSCRIPTION

Part No.	Description	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	____	$48.00	____	
	(if quantity is greater than 1 discount is 40%)			____	
	TOTAL DOLLARS for 5951-6111				____
5951-6112	COMMUNICATOR 2000	____	25.00	____	
	(if quantity is greater than 1 discount is 40%)			____	
	TOTAL DOLLARS for 5951-6112				____
5951-6113	COMMUNICATOR 3000	____	48.00	____	
	(if quantity is greater than 1 discount is 40%)			____	
	TOTAL DOLLARS for 5951-6113				____

□ BACK ISSUE ORDER FORM (cash only in U.S. dollars)
(subject to availability)

Part No.	Description	Issue No.	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	____	____	$10.00	____	
		____	____	10.00	____	
		____	____	10.00	____	
	TOTAL DOLLARS					____
5951-6112	COMMUNICATOR 2000	____	____	$ 5.00	____	
		____	____	5.00	____	
		____	____	5.00	____	
	TOTAL DOLLARS					____
5951-6113	COMMUNICATOR 3000	____	____	$10.00	____	
		____	____	10.00	____	
		____	____	10.00	____	
	TOTAL DOLLARS					____
TOTAL ORDER DOLLAR AMOUNT						____

□ SERVICE CONTRACT CUSTOMERS

You will receive one copy of either COMMUNICATOR 1000, 2000, or 3000 as part of your contract. Indicate additional copies below and have your local office forward. Billing will be included in normal contract invoices.

Number of additional copies ________

FOR HP USE ONLY

CONTRACT KEY

5951-6111 Number of additional copies ________

5951-6112 Number of additional copies ________

5951-6113 Number of additional copies ________

Approved ______________________

HEWLETT-PACKARD
COMMUNICATOR SUBSCRIPTION AND ORDER INFORMATION

The Computer Systems COMMUNICATORS are bi-monthly systems support publications available from Hewlett-Packard on an annual (6 issues) subscription.

The following instructions are for customers who do not have Software Service Contracts.

1. Complete name and address portion of order form.
2. For new direct subscriptions (see sample below):
   a. Indicate which COMMUNICATOR publication(s) you wish to receive.
   b. Enter number of copies per issue under Qty column.
   c. Extend dollars (quantity x list price) in Extended Dollars column.
   d. Enter discount dollars on line under Extended Dollars. (If quantity is greater than 1 you are entitled to a 40% discount.*)
   e. Enter Total Dollars (subtract discount dollars from Extended List Price dollars).

   **To qualify for discount all copies of publications must be mailed to same name and address and ordered at the same time.*

SAMPLE

☒ **DIRECT SUBSCRIPTION**

Part No.	Description	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	3	$48.00	$144.00	
	(if quantity is greater than 1 discount is 40%)			57.60	
	TOTAL DOLLARS for 5951-6111				$86.40

3. To order back issues (see sample below):
   a. Indicate which publication you are ordering.
   b. Indicate which issue number you want.
   c. Enter number of copies per issue.
   d. Extend dollars for each issue.
   e. Enter total dollars for back issues ordered.

   All orders for back issues of the COMMUNICATORS are cash only orders (U.S. dollars only) and are subject to availability.

SAMPLE

☒ **BACK ISSUE ORDER FORM** (cash only in U.S. dollars)
(subject to availability)

Part No.	Description	Issue No.	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	XX	1	$10.00	$10.00	
		XX	2	10.00	20.00	
				10.00		
	TOTAL DOLLARS					$30.00

4. Domestic Customers: Mail the order form with your U.S. Company Purchase Order or check (payable to Hewlett-Packard Co.) to:

   HEWLETT-PACKARD COMPANY
   Computer Systems COMMUNICATOR
   P.O. Box 61809
   Sunnyvale, CA 94088
   U.S.A.

5. International Customers: Order by part number through your local Hewlett-Packard Sales Office.

HEWLETT-PACKARD
COMPUTER SYSTEMS COMMUNICATOR ORDER FORM

Please Print:

Name ______________________________ Date ______________

Company ______________________________

Street ______________________________

City ____________________ State ____________ Zip Code ________

Country ______________________________

□ HP Employee Account Number ____________ Location Code ____________

□ DIRECT SUBSCRIPTION

Part No.	Description	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000		$48.00		
	(if quantity is greater than 1 discount is 40%)				
	TOTAL DOLLARS for 5951-6111				
5951-6112	COMMUNICATOR 2000		25.00		
	(if quantity is greater than 1 discount is 40%)				
	TOTAL DOLLARS for 5951-6112				
5951-6113	COMMUNICATOR 3000		48.00		
	(if quantity is greater than 1 discount is 40%)				
	TOTAL DOLLARS for 5951-6113				

□ BACK ISSUE ORDER FORM (cash only in U.S. dollars)
(subject to availability)

Part No.	Description	Issue No.	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000			$10.00		
				10.00		
				10.00		
	TOTAL DOLLARS					
5951-6112	COMMUNICATOR 2000			$ 5.00		
				5.00		
				5.00		
	TOTAL DOLLARS					
5951-6113	COMMUNICATOR 3000			$10.00		
				10.00		
				10.00		
	TOTAL DOLLARS					
TOTAL ORDER DOLLAR AMOUNT						

□ SERVICE CONTRACT CUSTOMERS

You will receive one copy of either COMMUNICATOR 1000, 2000, or 3000 as part of your contract. Indicate additional copies below and have your local office forward. Billing will be included in normal contract invoices.

Number of additional copies ________

FOR HP USE ONLY

CONTRACT KEY

5951-6111 Number of additional copies ______

5951-6112 Number of additional copies ______

5951-6113 Number of additional copies ______

Approved ____________________

HEWLETT-PACKARD
COMMUNICATOR SUBSCRIPTION AND ORDER INFORMATION

The Computer Systems COMMUNICATORS are bi-monthly systems support publications available from Hewlett-Packard on an annual (6 issues) subscription.

The following instructions are for customers who do not have Software Service Contracts.

1. Complete name and address portion of order form.
2. For new direct subscriptions (see sample below):
   a. Indicate which COMMUNICATOR publication(s) you wish to receive.
   b. Enter number of copies per issue under Qty column.
   c. Extend dollars (quantity x list price) in Extended Dollars column.
   d. Enter discount dollars on line under Extended Dollars. (If quantity is greater than 1 you are entitled to a 40% discount.*)
   e. Enter Total Dollars (subtract discount dollars from Extended List Price dollars).

   **To qualify for discount all copies of publications must be mailed to same name and address and ordered at the same time.*

SAMPLE

☒ DIRECT SUBSCRIPTION

Part No.	Description	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	3	$48.00	$144.00	
	(if quantity is greater than 1 discount is 40%)			57.60	
	TOTAL DOLLARS for 5951-6111				$86.40

3. To order back issues (see sample below):
   a. Indicate which publication you are ordering.
   b. Indicate which issue number you want.
   c. Enter number of copies per issue.
   d. Extend dollars for each issue.
   e. Enter total dollars for back issues ordered.

   All orders for back issues of the COMMUNICATORS are cash only orders (U.S. dollars only) and are subject to availability.

SAMPLE

☒ **BACK ISSUE ORDER FORM** (cash only in U.S. dollars)
(subject to availability)

Part No.	Description	Issue No.	Qty	List Price	Extended Dollars	Total Dollars
5951-6111	COMMUNICATOR 1000	XX	1	$10.00	$10.00	
		XX	2	10.00	20.00	
				10.00		
	TOTAL DOLLARS					$30.00

4. Domestic Customers: Mail the order form with your U.S. Company Purchase Order or check (payable to Hewlett-Packard Co.) to:

   HEWLETT-PACKARD COMPANY
   Computer Systems COMMUNICATOR
   P.O. Box 61809
   Sunnyvale, CA 94088
   U.S.A.

5. International Customers: Order by part number through your local Hewlett-Packard Sales Office.

Please photocopy this order form if you do not want to cut the page off. You will automatically receive a new order form with your order.

HEWLETT PACKARD

CONTRIBUTED SOFTWARE
Direct Mail Order Form

NOTE: No direct mail order can be shipped outside the United States.

Please Print:

Name ______________________ **Title** ______________

Company ______________________

Street ______________________

City ______________ **State** ______________ **Zip Code** ________

Country ______________________

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

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.

Domestic Customers: Cash required on all orders less than $50.00. Mail the order form with your check or money order (payable to Hewlett-Packard Co.) or your U.S. Company Purchase Order to:

HEWLETT-PACKARD COMPANY
Contributed Software
P.O. Box 61809
Sunnyvale, CA 94088

International Customers: Order through your local Hewlett-Packard Sales office. No direct mail order can be shipped outside the United States.

All prices domestic U.S.A. only. Prices are subject to change without notice.

ORDERING INFORMATION

Programs are available individually in source language on either paper tape, magnetic tape, or cassettes as indicated in the abstracts.

To order a particular program, it is necessary to specify the program identification number, together with an option number which indicates the type of product required. The program identification number with the option number composes the ordering number.

For example:

22113A-K01

The different options are.

K01 — Source paper tape and documentation
K21 — Magnetic tapes and documentation

NOTE

Specify 800 BPI or 1600 BPI Magnetic tape.

B01 — Binary tape and documentation
D00 — Documentation
L00 — Listing

Not all options are available for all programs.

Ten-digit numbers do not require additional option numbers such as K01, K21, etc. The 10-digit number automatically indicates the option or media ordered.

For example:

22681-18901 — The digits 189 indicate source paper tape plus documentation.
22681-10901 — The digits 109 indicate source magnetic tape plus documentation (800 BPI magnetic tape)
22681-11901 — The digits 119 indicate source magnetic tape plus documentation (1600 BPI magnetic tape)
22681-13301 — The digits 133 indicate source cassettes plus documentation

Only those options listed in each abstract are available.

Refer to the Price List for prices and correct order numbers.

Hewlett-Packard offers no warranty, expressed or implied and assumes no responsibility in connection with the program material listed.

HEWLETT-PACKARD
LOCUS CONTRIBUTED SOFTWARE CATALOG
DIRECT MAIL ORDER FORM

Please Print:

Name ____ Title __________________________

Company ____

Street ____

City ____ State ____________ Zip Code _______

Country ____

□ HP Employee Account Number _______ Location Code _________

Part Number	Description	Qty.	List Price Each	Extended Total
22000-90099	Locus Contributed Software Catalog		$15.00	

*If no sales tax is added, your state exemption number must be provided: #_________

If not, your order may have to be returned.

Your State & Local Sales Taxes*	
Handling Charge	1.50
TOTAL	

Domestic Customers: Mail the order form with your check or money order (payable to Hewlett-Packard Co.) to:

HEWLETT-PACKARD COMPANY
LOCUS CATALOG
P.O. Box 61809
Sunnyvale, CA 94088

International Customers: Order by part number through your local Hewlett-Packard Sales Office.

NOTE: No direct mail order can be shipped outside the United States. All prices domestic U.S.A. only. Prices are subject to change without notice.

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 ____________________ CUSTOMER REFERENCE # ____________

STREET ____________________ TAXABLE*? ____________

CITY ____________ STATE ____________ 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
P.O. Drawer #20
Mountain View, CA 94043

Phone: (415) 968-9200

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.

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

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

Printed in U.S.A. 3/79

Part No. 5951-6111