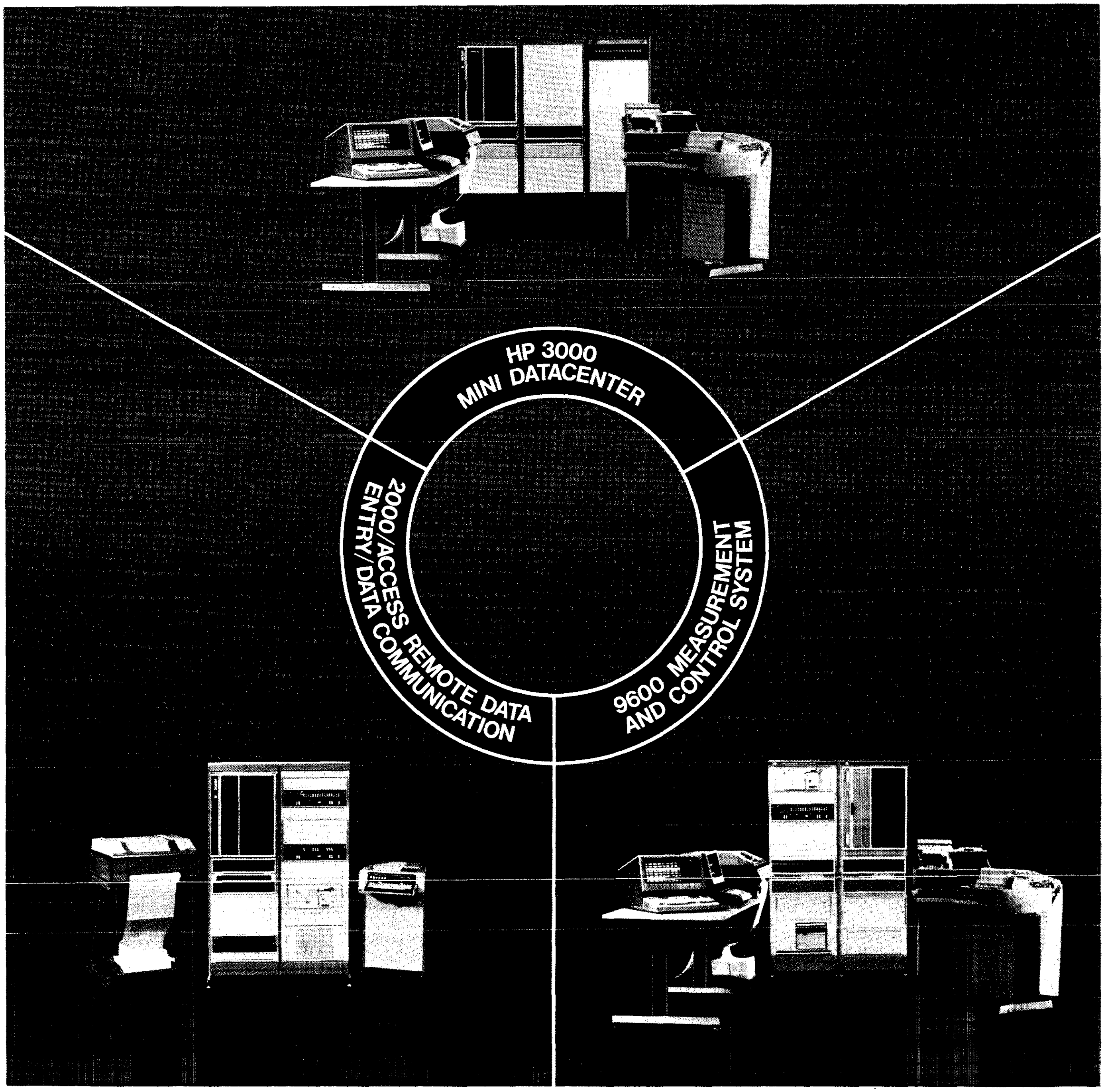
8/75

HEWLETT hp PACKARD

computer systems COMMUNICATOR

ISSUE 2
AUGUST 15, 1975



editor's note

This is the second issue of the newly launched Hewlett-Packard **Computer Systems Communicator.**

The **Communicator** is published by the support personnel of HP Data Systems, and brings you timely information on the HP 2000, 3000 and 9600/9700 Computer systems.

We hope you were able to find information of pertinent value in Issue No. 1 of the Communicator dated June 15, 1975. Our aim is to keep you up-to-date on all the currently available software and documentation as well as programming tips, training schedules, and other useful information.

This issue of the **Communicator** is divided into three parts. The first part with the light green border is for HP 2000 Computer Series users. The second section of the **Communicator** with the grey border is devoted to the HP 9600/9700 systems. The final section bordered in dark green contains data relative to HP 3000 Computer products.

The center section is reserved for a feature article and other items of general interest to HP users. The feature article in this issue describes the organization of our factory and field support people, including the functions they perform in supporting the many and varied customer installations throughout the world.

Your comments about the **Communicator** are welcome. Address correspondence to:

Editor
Computer Systems Communicator
HP Data Systems Division
Customer Engineering Department
11000 Wolfe Road
Cupertino, Ca. 95014

SUBSCRIPTION INFORMATION

The first three Preview Issues of the **Communicator** are being distributed without charge to a large list of potential subscribers.

Annual subscriptions (8 issues per year) beginning November 1975 will be offered as follows:

A. BASE SUBSCRIPTION. 1 copy per issue, $48.00/year.

B. ADDITIONAL SUBSCRIPTIONS. Extra copies to the BASE SUBSCRIPTION are available at the rate of $12.00/year for each additional copy, when ordered at the same time as the BASE SUBSCRIPTION and when copies are delivered to the same customer address as the BASE SUBSCRIPTION.

All HP customers with Software Service Contracts will be entitled to one BASE SUBSCRIPTION (1 copy per issue) at no additional charge. These customers may also purchase ADDITIONAL SUBSCRIPTIONS at the above price.

Complete instructions for ordering the **Communicator** will be published in the third Preview Issue.

contents

ABOUT THE HP 2000

Software Tips

Bulletins

Software Updates

ABOUT THE HP 9600/9700

Software Tips

Bulletins

Software Updates

GENERAL INTEREST

FEATURE ARTICLE

ABOUT THE HP 3000

Software Tips

Bulletins

Software Updates

software tips

This section of the Communicator contains timely how-to-do-it information contributed by HP customers and personnel. The information presented is pertinent to 2100/21MX based DOS and Timeshared systems.

You may find just the solution you've been looking for in this section, or merely a better way to accomplish a particular task. If you've found a solution on your own, why not send it to the Communicator so that everyone can share it.

DOS-III LOGICAL AND PHYSICAL DRIVERS

In the manual *DOS-III Data Communications Drivers,* part number 24307-90012, we describe how to use DOS-III logical and physical drivers in a data communications environment.

Some customers have stated that they could better understand how these physical/logical driver combinations work if we described the flow of control between the program they code and the physical and logical drivers.

This article discusses the flow of control in a logical/physical driver combination. The description is in 3 parts:

A. STANDARD CONTROL FLOW (for initialization and an I/O request)

B. PHYSICAL DRIVER CALLS TO LOGICAL DRIVERS

C. LOGICAL DRIVER RETURNS TO PHYSICAL DRIVERS

A. Standard Control Flow

When any logical/physical driver combination is used, a standard sequence of events occurs. Shown in the following paragraphs are the sequences for a logical/physical initialization and a typical I/O request.

Logical/Physical Driver Initialization

The following sequence occurs when the logical/physical driver combination is initialized. (Figure 1 illustrates the sequence)

a. The user's program executes a line initialization call to the appropriate logical initialization routine.

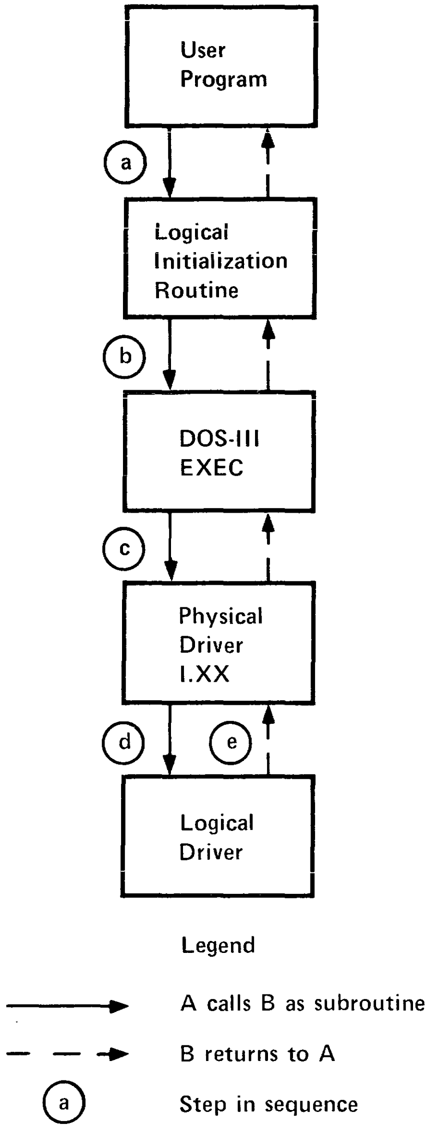


Figure 1. Logical/Physical Driver Initialization

This call must be made once for each device associated with the logical driver. For example, if the user's program uses logical driver A with 3 devices and logical driver B with 2 devices, it must make 3 calls to the logical initialization routine associated with logical driver A and 2 calls to the logical initialization routine associated with logical driver B.

b. The logical initialization routine executes an I/O EXEC control call (for "initialize") to the appropriate physical driver.

c. DOS-III passes control to the initiation section of the appropriate physical driver.

d. The physical driver makes a link between itself and the associated logical driver. When it has processed the initial parameters from the EQT Extension, the physical driver calls the associated logical driver.

e. The logical driver processes its own parameters, and returns.

Processing I/O Requests

The following sequence occurs when the user program issues a FORTRAN READ or WRITE, an I/O EXEC read, write, or control request. (Figure 2 illustrates the sequence.)

a. The user's program issues the I/O EXEC request (with or without wait).

a'. If the Terminal Control System (TCS) is being used, the call is made to TCS, which in turn issues the appropriate EXEC call on behalf of the user.

b. DOS-III passes control to the initiation section of the physical driver.

c. The physical driver passes control to the logical driver, specifying initiation.

d. The logical driver returns control to the physical driver, specifying either continuation or completion, after it has performed the actions necessary at request initialization.

e. The physical driver returns control to DOS-III (and from there, to the user program if without wait).

If the logical driver specifies completion in step d, the return to DOS-III is immediate. If the logical driver specifies continuation in step d, the physical driver is required to carry out a function (the action specified by the logical driver, and, in some cases, the action required by the I/O request). Then the physical driver returns control to DOS-III, specifying continuation (or, in the case of certain control functions, specifying completion).

f. If the physical driver starts an I/O operation in step e (continuation is specified), the resultant interrupts cause the hardware to pass control to the privileged interrupt subroutines in the physical driver.

g. When a character has been transmitted or received, or when it detects certain conditions, the physical driver passes control to the logical driver, specifying continuation. Note that in some cases, it may take several interrupts before the physical driver senses that a character actually has been transmitted or received.

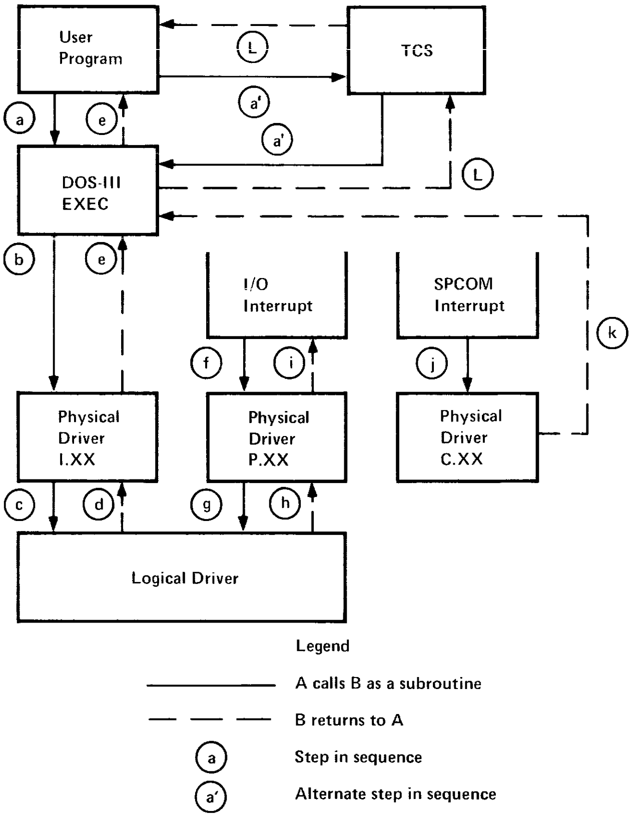


Figure 2. Processing I/O Requests

h. The logical driver determines why it has been called, takes appropriate action, and returns control to the physical driver, specifying either continuation or completion.

Note that the logical driver may require several intermediate I/O operations in order to execute the user I/O request. For example, it may have to transmit a control message, receive the response, and then transmit the data message.

When the logical driver detects the end of an intermediate I/O operation, it tells the physical driver to initiate the next intermediate I/O operation, through command bits returned to the physical driver.

i. If continuation is specified, the physical driver performs the action specified by the logical driver. If completion is specified, the physical driver calls the DOS-III privileged completion subroutine ($PCOM). In either case, the physical driver returns control to the interrupted program at the point where it was interrupted.

j. As soon as it is possible after the physical driver calls the privileged completion subroutine, DOS-III causes a "system interrupt" into the physical driver's completion section.

k. The completion section of the physical driver makes a completion return to DOS-III. If the user is not using TCS, and the request is without wait the user must detect the completion with a status call. DOS-III returns to the user if the call is made with wait.

l. If TCS is being used, it detects the completion and returns control to the user's program at the proper location depending upon whether the I/O request is issued with wait or without wait.

B. Physical Driver Calls To Logical Drivers

There are three calling sequences which serve as the interface between a physical driver and a logical driver: I/O Control, Read, and Write. They are discussed as separate topics below. For all three calling sequences, the following conventions apply:

a. The A-register contains status bits in the left byte (bits 11-15) and in the case of continuation Read calls, input data in the right byte (bits 0-7). Refer to Figure 3. Note: For an initial read call, a write call, or a call because a logical driver timing request has been completed, the right byte contains no data.

b. The B-register is negative for initiation calls and positive for continuation calls. Bits 0-14 of the B-register may be used by the physical driver for passing additional parameters to the logical driver.

c. Base page location 300 (octal) contains the address of the EQT entry associated with the current I/O operation. The physical driver always sets location 300 (octal) correctly before passing control to the logical driver.

d. The interrupt system is disabled.

Initiation calls from DOS-III through the I.xx entry point of the physical driver (with the exception of "Initialize" and "Clear" I/O Control requests) are passed immediately to the logical driver. Control passes from the physical driver to the logical driver by way of the read, write, and control entry points contained in the EQT extension. The physical driver specifies initiation by setting the B-register negative. ***The logical driver may reference any base page locations which are normally set by DOS-III at that time*** (for instance, the EQT locations).

The physical driver makes a continuation call to the logical driver whenever:

a. Character is transmitted or received.

b. Line error or break condition is sensed.

c. Logical driver timing request is completed.

The physical driver specifies continuation by setting the B-register positive. Since the physical driver is a privileged driver, the base page locations which are usually available to DOS-III drivers are not available to the logical driver for continuation calls (for instance, the EQT locations). However, the physical driver does set base page location 300 (octal) to the address of the EQT entry associated with the I/O request prior to calling the logical driver.

Contents														
15	14	13	12	11	10	9	8	7	6	5	4	3	1	0

a	b	c	d	e		input data	A-register
f	additional parameters						B-register

legend

A-register:

a = Time-out bit. If set to 1, a timing request initiated for the logical driver has been satisfied.

b = Line error bit. If set to 1, the communications line state or the interface PCA state is not correct for the operation requested.

c = Data error bit. If set to 1, incorrect parity or lost data has been detected.

d = Break bit. If set to 1, a break condition has been detected.

e = Special character bit. If set to 1, the input character *is not* a special character. (If set to 0, it may or may not be a special character.)

B-register:

f = Negative value initiation calls, positive for continuation calls.

Figure 3. Contents of A- and B-register (when control passes from physical to logical driver)

I/O Control Calling Sequence

The I/O Control calling sequence for passing control from a physical driver to a logical driver is as follows:

```
JSB    EQT24,I
```

(completion return from logical driver)
(continuation return from logical driver)
Note: EQT22, EQT23, etc. are locations in the EQT Extension buffer, presuming that word 1 of the EQT Extension is equivalent to word 18 of the EQT. For example, EQT24 is word 7 of the EQT Extension buffer.

If the logical driver returns control to the continuation address (P + 2) after an "Initialize" or "Clear" I/O Control request with the directive bits in the A-register clear, then the physical driver will evaluate the request.

Read Calling Sequence

The read calling sequence for passing control from a physical driver to a logical driver is as follows:

```
JSB    EQT22,I

(completion return from logical driver)
(continuation return from logical driver)
```

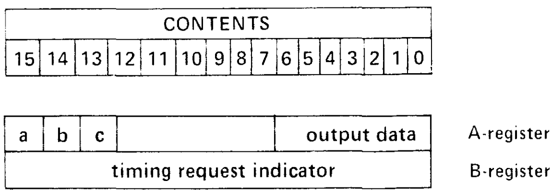
The physical driver will not initiate a read operation unless directed to do so by the logical driver. If the status bits in the A-register indicate a line error, a data error, or break condition, the logical driver should not consider the right byte of the A-register to be data.

Write Calling Sequence

The Write Calling Sequence for passing control from a physical driver to a logical driver is as follows:

```
JSB    EQT23,I

(completion return from logical driver)
(continuation return from logical driver)
```

The physical driver will not initiate a write operation unless directed by the logical driver to do so. The logical driver must always provide the first character in the right byte (bits 0-7) of the A-register when directing the physical driver to initiate a write operation.



legend

A-register:

a = **Start read bit. If set to 1, configure the interface and the communications line for a read operation; then start input.**

b = **Start write bit. If set to 1, configure the interface and the communications line for a write operation; then start output.**

c = **NOP bit. If set to 1, no operation is to be performed. Return to the calling or interrupted program.**

Note: **When a, b, and c are all clear (0), the logical driver is directing the physical driver to continue sending or receiving (whichever was previously specified), or to continue processing a control request.**

B-register: **A timing request indicator value**

<0 = the desired time interval, in 1/10 seconds.

=0 = no change

>0 = cancel current timing request

Figure 4. Contents of A- and B-registers (when control is passed from logical to physical driver)

C. Logical Driver Returns To Physical Drivers

There are two return locations to which control is passed by a logical driver: a completion return (P+1) and a continuation return (P+2), where P is the address of the JSB instruction in the physical driver which passed control to the logical driver. If the logical driver can complete the I/O request without further physical driver processing, the P+1 return is used; if the particular I/O request cannot be completely satisfied by the logical driver, the P+2 return is used.

When control returns by way of the continuation return (P+2), the logical driver may direct the physical driver to:

a. continue the current I/O operation;

b. start a new read or write operation; or

c. perform no operation.

The logical driver uses the A-register for this purpose (refer to figure 4). If all bits in the A-register are clear, the physical driver continues with the current I/O operation. If the "start read" or "start write" bit is set, the physical driver sets up the interface board and the communications line to perform the specified operation. The "no operation" directive provides the logical driver with a method for processing timeouts which do not require changing or continuing the I/O operation in progress.

The logical driver initiates or cancels timing requests using the B-register and the continuation return (P+2). To initiate or update a timing request, the logical driver sets the B-register to a negative number equal to the desired time interval in tenths of seconds. To cancel a timing request, the logical driver sets the B-register to zero.

Peter Baker
HP Cupertino

CONCATENATION OF STRINGS IN HP BASIC

What is concatenation? It is the putting together of two or more strings (i.e., words) to get one.

HP's BASIC was the first in the industry to allow a straight-forward string concatenation. Here is how it works.

The secret is in the substring feature. Suppose A$="APPLE". Using substring feature:

A$	= APPLE (the whole string)
A$(3)	= PLE (third character to the end)
A$(1,3)	= APP (first through third character)
A$(2,2)	= P (second through second character)

Substrings are legitimate strings and can be programatically assigned values. Knowing that, one can concatenate a string. Here is a sample program demonstrating how to do it.

```
10 DIM  A$(5),B$(5),C$(10)
20 LET  A$ = "APPLE"
30 LET  B$ = "SAUCE"
40 LET  C$(1,5) = A$
50 LET  C$(6,10) = B$
60 PRINT  C$
70 END
RUN
APPLESAUCE
```

Did you see how it worked? Statement 40 assigned the characters APPLE(A$) to characters 1 through 5 of C$. Statement 50 assigned the characters SAUCE to characters 6 through 10 of C$, so when all of C$ was printed in line 60, out came APPLESAUCE.

Things Which Can Go Wrong

1. If while concatenating, the dimension of the receiving string is exceeded, your program will be terminated.

2. Be sure that all internal characters of a string are defined, otherwise, a terminal error will result.

3. If the string being assigned to a substring is longer than the substring, truncation will result. This is different from 1 as the example shows:

```
10   DIM A$(72),B$(10)
20   LET B$= "APPLESAUCE"
30   LET A$ (1,5) = B$
40   PRINT A$
50   END
RUN
APPLE
```

4. If the assigned string is smaller than the receiving string then spaces will be filled in.

```
10   DIM A$(72),B$(10)
20   B$ = "APPLESAUCE"
30   A$(1,18) = B$
40   PRINT A$; "END"
50   END
RUN
APPLESAUCE        END
```

The substring feature is good for writing subroutines to do lots of other useful things. They include string to number conversion and vice versa, locating a string within a string and creating pseudo string arrays. Think about it.

Exercises: You must use string concatenation.

Write a program to:

1. Change the value of the string, B$="HOT APPLE SAUCE" to B$="HOT CHILE SAUCE." Print it out.

2. Change B$ = "HOT APPLE SAUCE" to "HOT TABASCO SAUCE." Print it out.

3. Write a program to print out the days of the week given A$="MONTUESWEDNESTHURSFRISAT-SUNDAY," no input, and one unassigned string variable.

Jean Danver
HP Cupertino

bulletins

2000/F TO 2000/ACCESS SYSTEM UPGRADE

The procedure for upgrading 2000/F Timeshared Data Systems to 2000/Access Systems is described in 2000/F to 2000/ACCESS System Upgrade Kit and Conversion Program Manual (HP19665-90001). Price is $2.00.

This manual consists of two sections. Section I describes the contents of the HP 19665 2000/F to 2000/Access Upgrade Kit and the installation procedures for the kit.

Section II describes the 2000/F to 2000/Access Conversion Program, including the conversion environment, user responsibilities, HP responsibilities, program availability, and operating instructions.

Anna Holland
HP Cupertino

software updates

HP 24343 I/O PROCESSOR

The I/O Processor (HP 24343) for the HP 2000F and HP 2000C' has been modified and carries Revision Number 1514, released April 15, 1975.

The revision contains the following modifications.

1) The line printer logic will now diagnose a power-off condition for the HP 2610 and HP 2614 Line Printers.

2) The line printer logic will now support the HP 2613 and HP 2618 Line Printers.

3) The IOP loader was modified so that it does not need to be manually halted after loading the IOP program. It will automatically halt.

4) Powerfail logic has been corrected to take care of several bugs, and has been modified to handle hardware changes in the 12920 Multiplexer.

   a) This modification is to allow the IOP to use the modified version B 12920 Multiplexer. However, this version of the IOP can also use version A of the 12920 Multiplexer.

The IOP has also been corrected to solve the problem with systems having port lockouts when using an HP 2767A Line Printer.

Peter Brickey
HP Cupertino

documentation

The following tables list all currently available software manuals, divided into the categories — DOS, Timeshare Systems, Languages, and other manuals. This list supersedes the previous list in the **Communicator**. Copies of manuals and update packages can be obtained from your local Sales and Service Office. The address and telephone number of the office nearest to you are listed in the back of all reference manuals.

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manuals you need on the Corporate Parts Center form supplied at the back of the **Communicator**. If you require an update package only send your request to:

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

DISC OPERATING SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02100-90074	HP 2605A Console Printer Driver	$ 1.00	3/72	
02116-91779	Moving-Head Disc Operating System	15.00	3/71	7/72
02762-90004	HP 2762A Terminal Printer Driver	1.00	5/73	
02767-90007	DOS/RTE 2767 Line Printer Driver	1.00	12/70	
02892-90005	HP 2892A Card Reader Driver	1.50	6/72	
12560-90023	DOS and RTE CALCOMP Plotter Driver	1.50	2/70	
12587-90011	HP 12587B Asynchronous Data Set Interface Driver Reference Manual	5.00	5/74	
12602-90023	DOS/RTE Mark Sense Drivers Kit 12602B	1.00	8/70	
12908-90004	HP 12908 Writable Control Store Driver	1.00	2/75	
12920-90004	HP 12920B Asynchronous Multiplexer Interface Driver Reference Manual	5.00	5/74	
13024-90010	DOS/DOS-M Magnetic Tape Unit Driver	1.00	4/72	5/73
24307-90006	DOS-III Reference Manual	20.00	2/75	
24307-90012	DOS-III Data Communications Drivers	7.50	11/74	1/75
24307-90018	DOS-III Pocket Guide	3.50	4/75	
24307-90002	DOS-III Terminal Printer Driver	1.00	1/75	
24307-90073	DOS-III Standard Drivers	6.00	1/75	
24376-90001	IMAGE/2000 Data Base Management System Reference Manual	11.00	11/74	
5951-1366	Cross Reference Table Generator	1.00	8/74	
5951-1375	Generating DOS-M	2.50	7/72	5/73
5951-1381	DOS-M/2000C Timeshared BASIC File Handler	1.00	5/71	
5951-1393	Generating DOS	2.00	9/71	
5951-1394	2000C File Interface for DOS-M	1.00	6/71	

TIMESHARED SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02000-90048	20856A Timeshared BASIC/2000, Level E, Reference Manual	$10.00	9/74	
02000-90049	20856A Timeshared BASIC/2000, Level E, System Operator's Manual	5.00	9/74	
02000-90073	20854A Timeshared BASIC/2000, Level F, Reference Manual	7.50	10/74	2/75
02000-90074	20854A Timeshared BASIC/2000, Level F, System Operator's Manual	10.00	6/75	
19665-90001	2000/F to 2000/ACCESS System Upgrade Kit and Conversion Program Manual	2.00	7/75	
5952-4491	20854A Timeshared BASIC/2000, Level F, Pocket Guide	0.15	10/74	

LANGUAGE MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02116-9014	HP Assembler Manual	$ 6.50	11/74	
02116-9015	HP FORTRAN Manual	5.00	3/74	
02116-9016	Symbolic Editor	4.50	2/74	
02116-9072	ALGOL Reference Manual	10.00	10/74	
02116-9077	20392A HP BASIC Reference Manual	15.00	9/74	
12907-90010	Implementing the HP 2100 Fast FORTRAN Processor	5.00	11/74	
22687-90009	Learning Timeshare BASIC	3.00	5/75	
24307-90014	DOS III Assembler Reference Manual	8.00	7/74	
5951-1321	HP FORTRAN IV Reference Manual	6.00	11/74	
5951-1377	Assembler, FORTRAN, and ALGOL Error Messages	1.00	4/71	

ADDITIONAL MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02000-90055	2000C/2000F IDF Author's Manual	$ 8.50	1/73	8/74
02000-90080	HP 2000E to HP 2000F Conversion Guide	1.00	12/73	
02022-90014	Magnetic Tape Reformatting System Support Utilities	1.50	1/74	
02100-90129	HP 2100 Microassembler Coding Form	5.00		
02100-90140	Decimal String Arithmetic Routines	6.50	10/73	
02108-90008	Microprogramming 21MX Computers Reference Manual	5.00	8/74	2/75
02116-91751	Prepare Tape System	2.50	8/74	
02116-91752	Magnetic Tape System	6.00	6/71	
02116-91780	2100 Series Relocatable Subroutines	11.00	10/73	
19655-90005	HP 19655B Management/260 Data System Preliminary Reference Manual	10.00	11/74	
19662-90001	Management/230 Data System Preliminary Reference Manual	5.00	6/74	
20308-90001	Instructional Management Facility Proctor's Manual	7.00	9/74	
20308-90003	Instructional Management Facility System Manager's Reference Manual	5.00	10/74	
20309-90001	Instructional Dialogue Facility Proctor's Manual	10.00	9/74	
20309-90003	Instructional Dialogue Facility Course Developer's Manual	6.00	8/74	
20309-90005	Instructional Dialogue Facility Author's Pocket Guide	3.50	10/74	
20310-90001	HP MATH Teacher's Handbook	5.00	9/74	
20310-90005	HP MATH Proctor's Manual	5.00	9/74	
20310-90007	HP MATH Curriculum Guide	20.00	7/74	
20311-90001	Timeshared Graphics for Tektronix Terminals	7.00	8/74	
20311-90003	Timeshared Graphics Plotting Package	25.00	6/74	
20352-90001	Educational Budget and Accounting System — System Overview	5.00	6/74	
20352-90002	Educational Budget and Accounting System Reference Manual	15.00	3/75	
20352-90003	Educational Budget and Accounting System — Technical Manual	75.00	3/75	
20353-90001	Educational Payroll System — System Overview	3.50	10/74	
24380-90001	HP 2100 Remote Job Entry Processor	3.00	10/73	
24383-90001	Course Writing Facility	15.00	5/74	
24384-90001	College Information System — System Overview	5.00	6/74	
24384-90003	College Information System Reference Manual	*		
24384-90005	College Information System Technical Manual	95.00	5/75	
24387-90001	Basic Analysis and Mapping Program Manual	12.00	6/74	5/75
24387-90002	Basic Analysis and Mapping Program Pocket Guide	10.00	6/74	
5951-1352	The Librarian	1.00	1/71	
5951-1353	Special Purpose Magnetic Tape Loader/CAI English	1.00	1/71	
5951-1371	HP 2100 Front Panel Procedures	1.00	8/73	
5951-1376	Basic Binary Loader/Disc Loader, Basic Moving-Head Disc Loader	1.00	4/74	
5951-1392	Magnetic Tape System	1.00	7/71	
5951-1397	Commercial Subroutines	15.00	8/71	

*Price to be determined.

training schedule

The schedule for training courses related to HP 2000 systems is presented below. Included are courses scheduled for the next four months (September through December).

Each issue of the Communicator will provide timely information on training to assist you in registering for classes applicable to your system.

MAINTENANCE COURSES

COURSE NUMBER	PRODUCT	COURSE LENGTH	DATES	LOCATION
22940A	2100A Operations & Maintenance	10 Days	Sept 8–19	Rockville
			Oct 6–17	Cupertino
			Nov 3–14	Rockville
			Dec 1–12	Cupertino
22941A	21MX Operations & Maintenance	5 Days	Sept 8–12	Cupertino
			Sept 29–Oct 3	Rockville
			Nov 17–21	Cupertino
			Dec 15–19	Rockville
22942A	7900A Operations & Maintenance	5 Days	Oct 20–24	Cupertino
22943A	7970B Operations & Maintenance	5 Days	Sept 15–19	Cupertino
22944A	7970E Operations & Maintenance	5 Days	Nov 3–7	Cupertino

SOFTWARE COURSES

COURSE NUMBER	PRODUCT	COURSE LENGTH	DATES	LOCATION
22952B	DOS IIIB	5 Days	Nov 3–7	Cupertino
22953A	2100 TCS/IMAGE	5 Days	Nov 10–14	Cupertino
22973A	2000/ACCESS DATA ENTRY, File Management & RJE	5 Days	Sept 29–Oct 3	Cupertino
			Dec 1–5	Cupertino
22950A	2100 Minicomputer Assembler	5 Days	Sept 22–26	Rockville
			Sept 29–Oct 3	Cupertino
			Nov 17–21	Cupertino
			Dec 1–5	Rockville
22959A	21MX Minicomputer Assembler	5 Days	Sept 8–12	Rockville
			Oct 20–24	Cupertino
			Oct 20–24	Rockville
			Nov 17–21	Rockville
			Dec 8–12	Rockville
22960A	21MX Microprogramming	5 Days	Sept 22–26	Cupertino
			Oct 27–31	Cupertino
22974A	Minicomputers in Manufacturing	2 Days	Oct 16–17	Cupertino

HP Training Centers

Training is conducted in the U.S.A. at facilities in Cupertino, California and Rockville, Maryland.

Each Training Center is staffed with professional instructors. Courses are designed such that the student will receive both classroom instruction and practical, hands-on experience. By attending the courses in the recommended sequence for your particular HP system, the student will gain the most beneficial training available to meet the needs of your specific application.

Western Training Center

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

Eastern Training Center

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

Accommodations

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

Registration

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

Cancellations

In the event you find you cannot attend a particular class we would appreciate your prompt notification.

software tips

This section of the Communicator contains timely how-to-do-it information contributed by HP customers and personnel. The information presented is pertinent to HP 9600/9700 systems.

You may find just the solution you've been looking for, or merely a better way to accomplish a particular task. If you've found a solution on your own, why not send it to the Communicator so that everyone can share it.

A LOOK AT HP FORTRAN IV OBJECT CODE GENERATION

How many times have you had the need to examine the assembly code produced by a high level language such as FORTRAN or ALGOL, or how often has your curiosity alone taken you there? In solving a problem a programmer usually selects the language in which he/she feels most competent. The experienced programmer usually has a firm understanding of code generation, and is thus able to accurately and quickly formulate a reason for using one statement rather than another (such as using a two branch IF statement versus a three branch).

However, in some cases the programmer using high level languages is not concerned with code optimization on levels other than the one chosen to solve the problem. This article describes how assembly language code is generated by certain statements in the HP FORTRAN IV Compiler and suggests hints for optimization on the source level. Hopefully this will give you a better idea of high level language code generation and a predictable means for selecting the appropriate language.

The universal problem for a computer manufacturer today is marrying two problems:

1. A good algorithm for the compiler writer.
2. Instruction set compatible with algorithm.

The FORTRAN user is concerned basically with four distinct factors, all of which influence turnaround time and job cost:

1. Compiler size (4K,10K,etc.).
2. Compiler speed.
3. Compiler generated object code.
4. Speed of compiler generated object code.

The size (and sometimes speed) of the compiler is of interest to the FORTRAN user during development and debugging stages of a program. Problems can develop depending upon the environment in which the program is compiled (batch, real-time, etc.). If the compiler is relatively large, a FORTRAN job might find itself at the bottom of the scheduling queue competing for memory space in a real-time system or virtually crawling in a batch system due to its many segments and overlays. While both size and speed of the FORTRAN compiler are important factors during program development, the user typically has little or no control over these factors. An exception is the HP Real-Time Executive System, in which the FORTRAN Compiler can be given highest priority.

Most language translators have no compile-time options which directly relate to object code optimization; i.e., they apply a fixed set of optimizations to all programs. The number and type of optimizations performed by compilers are many and varied, but they all can be classified into two categories:

1. Machine independent optimizations.

2. Machine dependent optimizations.

In each category there are optimizations for speed and optimizations for size of the object program although optimizations for speed versus size are almost always in conflict.

In the absence of a definitive document describing precisely the set of all optimizations performed by a given compiler, it is extremely difficult to determine conditions under which optimizations are done or should be done. Thus, I will display only a few significant examples in which the FORTRAN user can manually perform at the source code level to affect object code performance.

Raising A Number To A Whole Power

When programming a large or complex application, it is often tempting to minimize FORTRAN source code thinking that this will also minimize the object code. This is not always true. A common example is the following:

```
                  EXAMPLE 1A
FORTRAN         OBJECT CODE
Z=4             JSB     .RTOR
R=X**Z          DEF     X

                DEF     R.001     (Address of 4.)
                JSB     ERRO      (Jump to error
                JSB     .DST       routine)
                DEF     R
```

The routine .RTOR must go through an algorithm to achieve the correct result. However, if we do a series of multiplications, we can achieve the same result (without worrying about a negative exponent) and save the execution of ERRO:

```
                  EXAMPLE 1B
 FORTRAN          OBJECT CODE
R=X*X*X*X*       JSB     .DLD

                 DEF     X
                 JSB     .FMP      X*X
                 DEF     X
                 JSB     .FMP      X*X*X
                 DEF     X
                 JSB     .FMP      X*X*X*X
                 DEF     X
                 JSB     .DST
                 DEF     R
```

Note: Assume .DLD,.FMP,.DST, are replaced with DLD,FMP, AND DST instructions at load time.

In example IA, an additional jump to an error routine is generated, which checks for the range of Z. This could be very costly if the routine were executed quite frequently.

Mixed Mode Expressions

This is an area which is not obvious to everyone. The earlier compilers were not designed to continue with mixed mode expressions without aborting the compilation. In those days, high-level languages were simple minded and very strict syntactically. Today, software sophistication has eliminated some of that, but at the same time introduced a new variable - compiler interpretation. Just what does the compiler do under a given circumstance? Most modern-day compilers will accept mixed expressions such as

K=A+1 or Z(7)=16/A**2

but the object code results can be costly.

```
                EXAMPLE 2A
FORTRAN     OBJECT CODE
 A=A+7      LDA    C.0001    (Address of 7)
            JSB    FLOAT     (Charge to F.P.)
            JSB    .FAD
            DEF    00020     (Address of right A)
            JSB    .DST
            DEF    00020
```

```
                   EXAMPLE 2B
FORTRAN      OBJECT CODE
 A=A+7.      JSB    .DLD
             DEF    C.001     (Address of 7.)
             JSB    .FAD
             DEF    00020
             JSB    .DST
             DEF    00020
```

Notice the elimination of the jump of FLOAT in Example 2B. Imagine if the routine in Example 2A were inside a DO-Loop executed 10,000 times? Why not do it on an HP-65 hand held calculator – it might be faster!

Data Conversion

One of the most overlooked areas of source code optimization is minimizing the number of internal data conversions during execution. The following example will illustrate that the most elegant algorithm may not be the most efficient:

EXAMPLE 3A

```
    FORTRAN              OBJECT CODE

    DO 4 J=1,1000             LDA   03761   (Address of 1)
 4  X(J)=J                    STA   I       (Save first
                                             time in)
                        LOOP  LDA   I
                              ADA   03755   (Add-1)
                              ALS           (Array ele-
                                             ment address)
                              ADA   00004   (Compute cur-
                                             rent address)
                              STA   A.001

                              LDA   03756   (Address of 4)
                           →  JSB   FLOAT   (Convert to
                                             F.P.)
                              JSB   .DST    (Store in cur-
                                             rent array
                                             element
                                             address)
                              DEF   A.000,I
                              LDA   I
                              ADA   03761   (Increment
                                             index
                                             pointer)
                              STA   I
                              CMA,  IMA
                              ADA   03752   (Address of
                                             1000)
                              SSA,  RSS
                              JMP   LOOP
                              (Continue Processing)
```

The above will result in 1000 conversions of J from integer to real at execution time. The following code in Example 3B, while more lengthy in its source form, will execute faster. For most compilers (including HP's) it will take less memory space because no conversions are required.

EXAMPLE 3B

```
    FORTRAN              OBJECT CODE

    Z =0.0                    JSB   .DLD    (Address of 4.0)
    DO 4 J=1,1000             DEF   00351
 4  X(J)=Z                    JSB   .DST    (Store in B)
                              DEF   B
                              LDA   00361   (Save Index)
                              STA   I
                        LOOP  LDA   I
                              ADA   00356
                              ALS
                              ADA   00004
                              STA   A.001
                              JSB   .DLD
                              DEF   B
                              JSB   .DST
                              DEF   A.001,I
                              LDA   I
                              ADA   00361
                              STA   I
                              CMA,  INA
                              ADA   00355
                              SSA,  RSS
                              JMP   LOOP
                              (Continue Processing)
```

Notice that the same amount of object code is generated for both examples, but during execution, Example 3B shows a significant cut in execution time due to the elimination of the subroutine float.

Thus, it would be wise for the user of a FORTRAN compiler to determine how the compiler handles constants with respect to data type. It would be easier for a compiler to ignore the context in which a constant is used and simply preserve the type specified by the user rather than to perform a compile-time conversion of the constant to the type most optimum for execution. For example,

```
IF(X.EQ.17)K=2.0
```

A "quick and dirty" compiler might omit object code which contains conversions of 17 from INTEGER to REAL and of 2.0 from REAL to INTEGER. If such is the case, the user should always attempt to match the constant type with the type of usage, which for the above example would become:

```
IF(X.EQ.17.)K=2
```

Expression Evaluation

The user should always attempt to minimize the number of execution-time evaluations of an expression. Unless the FORTRAN user is designing a generalized or special purpose software package for common use, object code optimization is usually at the bottom of the think list. Although, whatever your purpose might be for using FORTRAN, the following general rules will typically apply:

1. Never use a REAL variable or constant when an INTEGER will suffice.

2. Never use a DOUBLE PRECISION variable when a REAL variable will suffice.

3. To save core space, use EQUIVALENCE statements to effect overlay of arrays when possible.

4. Manually "Unroll" DO-Loops such as in Examples 3A and 3B.

In conclusion, many of the recommendations set forth in the previous section seem to be defeating the purpose of a high-level language, which is to free the user from restrictions, details of machine characteristics, etc. While a language unencumbered by restrictions may be easier to use than a rigid one, it is usually more difficult for a compiler to optimize. Until compiler optimizations for a given language become extremely well refined and universally applied, the user who is truly concerned about object code performance must be willing to share the optimization burden.

Larry W. Smith
HP Cupertino

RTE-B MEMORY REQUIREMENTS

The following list summarizes the number of memory locations required for storing specific BASIC program commands:

COMMAND	NO. OF REPORT LOCATIONS (Decimal)
REM	5
DIM A (10)	31
DIM B (100)	209
LET A=1	10
FOR I = 1 to 5 NEXT I	17
READ A (1) DATA 1.5	14
IF A=2 GO TO 100	11
PRINT	4
PRINT #6	7
GOSUB 1000 RETURN	10
END	2
TIME (T)	65
TRNON (200,120000)	12
SETP (200,1)	12
START (200,10)	12
DSABL (200)	8
ENABL (200)	8
AISQV (1,1,V,E)	14
AISQV (1,1,V(1),E)	47
C(1) = 1	35
AIRDV (1,C(1),V(1),E)	22
PACER (1,5,2)	16
SGAIN (50,1000)	12
RGAIN (50,G)	13
AOV (1,1,5.7,E)	13
AOV (1,1,V,E)	9
RDBIT (4,2,B)	17
RDWRD (4,A)	13
WRBIT (3,5,1)	16
WRWRD (3,A)	10
SENSE (1,1,1,1)	20
IOR (M,N,R)	19
INOT (A,B)	14
IEOR (M,N,R)	10
IAND (M,N,R)	10
ISHFT (M,N,R)	18
IBTST (V,B,S)	19
IBTST (M,N,R)	9 *
IBCLR (M,N,R)	10
ISET C("17077",K)	13
LET A=ABS (-7)	14
LET A=EXP (A)	8
LET A=INT (A)	6
LET A=LOG (A)	10
LET B=LOG (A)	11
LET B=RND (A)	8
LET B=SGN (A)	8
LET B=SQR (A)	8
LET A=SWR (2)	10
PRINT TAB (A);A	10
LET A=SIN (B)	8
LET A=COS (B)	8
LET A=TAN (B)	8
LET A=OCT (B)	8
REM THIS STATEMENT USES UP A LOT OF MEMORY	25

*Same statement as above, but with symbols previously defined.

In some instances a given statement will take somewhat more or less memory depending upon whether new symbols are being defined by the statement or whether new values are being assigned to previously defined symbols. During the execution phase of BASIC, the system will temporarily require more memory than that occupied by a dormant program.

Joe Diesel
HP Cupertino

CENTRAL INTERRUPT REGISTER

The Central Interrupt Register provides the capability to determine which select code has been interrupted. This feature was first implemented on the 2116B computer. The primary use of the Central Interrupt Register is illustrated by the following portion of the $CIC routine used by the RTE system.

```
$CIC     NOP
         CLF 0          turn off interrupts
         STA XA,I       save registers
         STB XB,I
         ERA, ALS
         SOC
         INA
         STA XEO,I
         LIA 4          load interrupted
                        select code
```

A JSB $CIC,I is contained in each select code except those select codes using privileged interrupt. The implementation of the JSB indirect instruction insures the CLF 0 will hold off additional interrupts. Additional system code is then executed resulting in control being transferred to the proper driver required to service the interrupt. The use of the Central Interrupt Register reduces system design complexity and thus reduces amount of system code required to service interrupts.

George Taylor
HP Cupertino

RTE/BATCH SPOOL MONITOR

Referencing the System Input Device in Spooled Jobs

The following spooled job, where source is expected from the system input device, will not work:

```
*RU,JOB,1
;:JO
;:LG,1
;:RU,FTN4,5,99
;:RU,LOADR,99
;:RU,10G
;:EO

CNTRL-D
```

Why?

Because LU5 is transformed by the system to a spool file. The assembler tries to read its input from LU5, which is now a spool file and does not contain the source code. Meanwhile back at the device known as LU5, the input sits unnoticed. Also, if the program expects data input from the system input device, it won't get it by issuing reads to LU5.

References to LU5 will work only if the source code and data are on the system input device with the job, because code and data will also be spooled.

A solution to the problem requires that some other LU be used to reference the system input device:

```
*RU,JOB,1
;:JO
;:LU,10,5          assigns LU10 = system input
                      device
;:RU,FTN4,10,99    command references LU10
;:RU,LOADR,99
;:RU,10G           programmer must hard code
                   I/O requests to LU10 instead
;:EO               of LU5
```

Hal Sindler
HP Cupertino

USING EXTENDED DATA CONTROL BLOCK BUFFERS

Data transfer between a program and a file can be speeded up if the packing buffer in the file's data control block (DCB) is made larger than the default value, which is 128 words (one block). For example, assume two files exist, FILE1 and FILE2, and both are type 3, 16 blocks in length and are physically adjacent on the disc.

```
                  .
                  .
                  .
        DIMENSION IDCB1 (656),IDCB2(656),IBUF(128)
                  .
                  .
                  .
        I=    (PACKING BUFFER SIZE)
        CALL OPEN(IDCB1,IER,NAME1,0,0,0,I)
        CALL OPEN(IDCB2,IER,NAME2,0,0,0,I)
  10    CALL READF(IDCB1,IER,IBUF,128,L)
        IF(L.EQ.-1) GO TO 20
        CALL WRITF(IDCB2,IER,IBUF,128)
        GO TO 10
  20    CALL WRITF(IDCB2,IER,IBUF,-1)
                  .
                  .
                  .
```

The total transfer time is approximately 1.20 seconds for I = 128 and 0.70 seconds for I = 640 (see Table 1). The larger the buffer size the faster the transfer rate, subject to the following constraints:

For actual data transfers, the system determines a buffer size that is:

- a multiple of 128 words long, and
- less than or equal to the packing buffer size specified in I (the buffer size parameter), and
- can be evenly and exactly divided in the total file size.

To make optimum use of system buffering, the DCB buffer area should be large enough to satisfy these constraints. For example, if your file has 50 blocks of 128 words each (file size = 6400 words) your DCB packing buffer could be 128 words, 256 words, 640 words, and so forth. No advantage is gained by a file size of 384 words (3X128) or 512 words (4 X 128) because the sizes do not divide evenly into 6400 words. Even if you specified these sizes, the system would use only 256 words for data transfer. Note that 16 words will always be added to the packing buffer for the DCB control information. The buffer size actually used by the system can be determined with the call, IDCBS.

Whenever a DCB size greater than 144 (16 + 128) is dimensioned you must inform the system with the optional parameter IDCBS in the CREAT or the OPEN calls.

```
THE FOLLOWING TRANSFERS ARE FROM A 480 BLOCK FILE

               1 BLOCK TRANSFER

    1 BLOCK BUFFER - TIME   =  0: 0:   .09
    2    "      "      "    =  0: 0:   .09
    3    "      "      "    =  0: 0:   .09
    4    "      "      "    =  0: 0:   .09
    6    "      "      "    =  0: 0:   .09
    8    "      "      "    =  0: 0:   .12
   12    "      "      "    =  0: 0:   .12
   16    "      "      "    =  0: 0:   .13
   24    "      "      "    =  0: 0:   .14

               2 BLOCK TRANSFER

    1 BLOCK BUFFER - TIME   =  0: 0:   .16
    2    "      "      "    =  0: 0:   .11
    3    "      "      "    =  0: 0:   .11
    4    "      "      "    =  0: 0:   .11
    6    "      "      "    =  0: 0:   .12
    8    "      "      "    =  0: 0:   .15
   12    "      "      "    =  0: 0:   .15
   16    "      "      "    =  0: 0:   .15
   24    "      "      "    =  0: 0:   .16
```

```
          4 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0:  .32
  2   "      "      "      =  0: 0:  .21
  3   "      "      "      =  0: 0:  .24
  4   "      "      "      =  0: 0:  .16
  6   "      "      "      =  0: 0:  .16
  8   "      "      "      =  0: 0:  .18
 12   "      "      "      =  0: 0:  .19
 16   "      "      "      =  0: 0:  .19
 24   "      "      "      =  0: 0:  .20
```

```
          8 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0:  .62
  2   "      "      "      =  0: 0:  .41
  3   "      "      "      =  0: 0:  .41
  4   "      "      "      =  0: 0:  .32
  6   "      "      "      =  0: 0:  .29
  8   "      "      "      =  0: 0:  .27
 12   "      "      "      =  0: 0:  .28
 16   "      "      "      =  0: 0:  .28
 24   "      "      "      =  0: 0:  .29
```

```
          16 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0: 1.21
  2   "      "      "      =  0: 0:  .81
  3   "      "      "      =  0: 0:  .84
  4   "      "      "      =  0: 0:  .67
  6   "      "      "      =  0: 0:  .56
  8   "      "      "      =  0: 0:  .51
 12   "      "      "      =  0: 0:  .52
 16   "      "      "      =  0: 0:  .46
 24   "      "      "      =  0: 0:  .47
```

```
          32 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0: 2.38
  2   "      "      "      =  0: 0: 1.62
  3   "      "      "      =  0: 0: 1.63
  4   "      "      "      =  0: 0: 1.40
  6   "      "      "      =  0: 0: 1.18
  8   "      "      "      =  0: 0: 1.04
 12   "      "      "      =  0: 0: 1.05
 16   "      "      "      =  0: 0:  .93
 24   "      "      "      =  0: 0:  .93
```

```
          64 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0: 4.82
  2   "      "      "      =  0: 0: 3.26
  3   "      "      "      =  0: 0: 3.21
  4   "      "      "      =  0: 0: 2.84
  6   "      "      "      =  0: 0: 2.33
  8   "      "      "      =  0: 0: 2.07
 12   "      "      "      =  0: 0: 2.08
 16   "      "      "      =  0: 0: 1.87
 24   "      "      "      =  0: 0: 1.77
```

```
          128 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0: 9.78
  2   "      "      "      =  0: 0: 6.57
  3   "      "      "      =  0: 0: 6.56
  4   "      "      "      =  0: 0: 5.72
  6   "      "      "      =  0: 0: 4.68
  8   "      "      "      =  0: 0: 4.16
 12   "      "      "      =  0: 0: 4.11
 16   "      "      "      =  0: 0: 3.76
 24   "      "      "      =  0: 0: 3.56
```

```
          256 BLOCK TRANSFER

  1 BLOCK BUFFER - TIME    =  0: 0:19.54
  2   "      "      "      =  0: 0:13.11
  3   "      "      "      =  0: 0:13.09
  4   "      "      "      =  0: 0:11.37
  6   "      "      "      =  0: 0: 9.35
  8   "      "      "      =  0: 0: 8.30
 12   "      "      "      =  0: 0: 8.20
 16   "      "      "      =  0: 0: 7.50
 24   "      "      "      =  0: 0: 6.99
```

```
          480 BLOCK TRANSFER
      "      "      "
  1 BLOCK BUFFER - TIME    =  0: 0:36.62
  2   "      "      "      =  0: 0:24.60
  3   "      "      "      =  0: 0:24.51
  4   "      "      "      =  0: 0:21.53
  6   "      "      "      =  0: 0:17.53
  8   "      "      "      =  0: 0:15.56
 12   "      "      "      =  0: 0:15.36
 16   "      "      "      =  0: 0:14.06
 24   "      "      "      =  0: 0:13.05
```

The above table was derived by executing the following program for DCB buffer sizes of 1,2,3,4,5,6,8,12,16, and 24 blocks to transfer files of 1,2,4,8,16,32,64,128,256, and 480 blocks. The time required to transfer the contents of FILE1 to FILE2 was computed for each complete file transfer.

```
FTN,L
      PROGRAM TEST
      DIMENSION NAME1(3),NAME2(3),IDCB1(3088),IDCB2(3088),IBUF(628),I(5)
      DIMENSION ITHEN(5),INOW(5)
      DATA NAME1/2HFI,2HLE,2H1 /,NAME2/2HFI,2HLE,2H2 /
      CALL RMPAR(I)
      CALL OPEN(IDCB1,IER,NAME1,0,0,0,I(1))
      CALL OPEN(IDCB2,IER,NAME2,0,0,0,I(1))
      CALL EXEC(11,ITHEN)
100   CALL READF(IDCB1,IER,IBUF,128,L)
      IF(L.EQ.-1) GO TO 300
      CALL WRITF(IDCB2,IER,IBUF,L)
      GO TO 100
300   CALL EXEC(11,INOW)
      CALL TIME(ITHEN,INOW,SEC)
      WRITE(6,400)I(3),INOW(4),INOW(3),SEC
400   FORMAT("       ",I3,"BLOCK BUFFER - TIME = ",2(I2,1H:),F5.2)
      CALL CLOSE(IDCB1,IER)
      CALL CLOSE(IDCB2,IER)
      END

      SUBROUTINE TIME(ITHEN,INOW,SEC)
      DIMENSION INOW(5),ITHEN(5),IBASE(4)
      IBASE(1)=100
      IBASE(2)=60
      IBASE(3)=60
      IBASE(4)=24
      DO 200 J=1,4
      INOW(J)=INOW(J)-ITHEN(J)
      IF(INOW(J)) 100,200
100   INOW(J)=INOW(J)+IBASE(J)
      INOW(J+1)=INOW(J+1)-1
200   CONTINUE
      SEC=INOW(2)+INOW(1)/100.
      END
      END$
```

Erryl Johnson
HP Cupertino

bulletins

NEW PRODUCTS FOR RTE USERS

RTE-III

HP has recently announced a new member of its family of Real-Time operating systems, RTE-III. This new software is able to take full advantage of the memory expansion capabilities of the 21MX. In addition, RTE-III (as well as RTE-II) can now support both the 7900A and 7905A disc drives, (even in the same system) and multi-terminal, DISC-BASED Real-Time BASIC.

RTE-III includes the following major new features:

1. Memory expansion capability up to 256K words.
2. Up to 64 partitions for disc-resident programs. Each partition can range in size from 2K words to a maximum of 16-18K. RTE-II allows only two partitions for disc-resident programs.

RTE-III supports all the same features as RTE-II, such as Input/Output Spooling, Multi-Stream Batch, and the Multi-Terminal Monitor. In fact, to the application programmer, RTE-II and RTE-III are identical - the difference is that RTE-III allows him up to 64 possible partitions for his program. (The exception to this is privileged drivers, which now must save the memory maps).

Since RTE-III requires Dynamic Mapping and uses the new 21MX instructions, it requires a 21MX. Minimum system is 32K words.

Real–Time Basic

Also being announced at this time is Real-Time disc-based BASIC. A superset of RTE/B, Real-Time BASIC also includes string capabilities and a file system similar to that on HP time-sharing systems.

Video Monitor Interface

Another recent announcement is the Video Monitor Interface. This card (HP 91200A) allows RTE systems to interface directly to standard commercial TV monitors. This card is fully compatible with U.S. and European television systems. With 3 cards it can even work in color. Up to five monitors can be daisy-chained together from the same 91200A interface. Industry-standard synchronization in the composite video signal provides for lock-on of studio TV equipment, allowing split-screen effects or mixing of TV card video with TV pictures.

Software available with the card includes an RTE driver, diagnostic and verification programs, and a graphics library of seven routines callable from FORTRAN IV, ALGOL, Real-Time BASIC and Assembly language.

9600MX Integration

HP is also announcing the integration of the M/30 Processor, the 12990A Memory Extender, 12976A Dynamic Mapping System, the 91200A Video Monitor, and the 7905A Disc into all the 9600MX systems. See your HP sales representative for further information.

Dave Sanders
HP Cupertino

POWER FAIL/AUTO RESTART FOR RTE-B

Automatic restart capability for Real Time BASIC is now available. To incorporate this into an RTE-B system requires regenerating the system with the addition of these four modules:

Module	HP Part Number	
RTC	92000-80003 92000-60003	(Source) Rev. B (Relocatable)
$ALC	92000-18002 92000-16002	(Source) (Relocatable)
DVP 43	92001-18004 92001-16004	(Source) Rev. C (Relocatable) Rev. C
AUTOR	92000-18001 92000-16001	(Source) (Relocatable)

Additionally, a modified generator should be ordered:

RTSGN	29101-60007	Rev. B

The modules RTC and $ALC compose RTBOS. If the RTBOS module used for generating the system is HP part No. 92000-60001, these two new modules replace it.

The generation procedure is very much like the usual procedure with these additions:

1. DVP 43 is relocated along with standard device drivers.
2. An equipment table number and select code must be assigned to DVP 43.
3. A logical unit number must be allocated corresponding to the EQT assignment in item 2.

4. The entry,

   4,ENT,$POWR

   must be incorporated into the interrupt table.

5. One more ID segment must be called for to accommodate the program AUTOR.

6. AUTOR must be relocated as a user program.

When a power failure and subsequent restart occur, a brief message (including the time when power failed) is printed on the system console.

Joe Diesel
HP Cupertino

software updates

HP 92002-16006 BATCH MONITOR LIBRARY

The IDCBS routine in the current Batch Monitor Library (92002-16006) does not function correctly but always returns a value of -11, meaning DCB is not open to the calling program. It should return the size of the DCB buffer if the DCB is open. The size is a multiple of 128 + 16 for type 3 files and above, and 16 for type 0 and 1.

The following routine is a temporary solution to the problem until a supported update of the Batch Monitor Library is released. It can be treated as a subroutine when loaded with the calling program:

```
ASMB,R,L,C
      NAM IDCBS,7
      EXT .ENTR
      ENT IDCBS
*
IDCB  NOP
IDCBS NOP
      JSB .ENTR       FETCH PARAM ADDR
      DEF IDCB
      LDB IDCB        GET THE OPEN FLAG
      ADB D9           FROM WORD 9 OF THE DCB
      LDA B,I
      CPA XEQT        IS THIS FILE OPENED?
      JMP OPEND        YES
      LDA MD11         NO, ERROR -11
      JMP EXIT
*
OPEND ADB MD7         BACK UP TO WORD 2
      LDA B,I
      SZA             FILE TYPE 0?
      CPA D1          FILE TYPE 1?
      CLA,RSS
      RSS
      JMP  RTNOK      YES 0 OR 1
*
*
      ADB D5          ADVANCE TO WORD 7
      LDA B,I         GET SIZE WORD
      ARS,ALR          BUT CLEAR BITS 0 AND 15
RTNOK ADA D16         ADD 16 TO BUFFER SIZE
EXIT  JMP IDCBS,I     RETURN DCB SIZE IN A
*
A     EQU 0
B     EQU 1
XEQT  EQU 1717B
D1    DEC 1
D2    DEC 2
D5    DEC 5
D9    DEC 9
D16   DEC 16
MD11  DEC -11
MD7   DEC -7
*
      END
      ENDS
```

Erryl Johnson
HP Cupertino

RTE-II

The following tables describe currently available software for the Real-Time Executive Disc-Based System (RTE-II).

PART NUMBER/ REVISION CODE	DESCRIPTION
20802-60001-C	System Dump used with fixed head disc only.
20805-60001-C	RTE Editor
20875-60001-E through 20875-60005-E	RTE FORTRAN II
24016-60001-A	Prepare Tape System
24129-60001-C	RTE/DOS ALGOL
24129-60002-C	RTE/DOS ALGOL
24151-60001-D	RTE/DOS Relocatable Library (EAU)
24152-60001-C	RTE/DOS FORTRAN IV Library
24153-60001-C	RTE-DOS FORTRAN Formatter
24170-60001-C through 24170-60003-C	RTE/DOS FORTRAN IV Compiler (5K BG area)
24177-60001-B	RTE/DOS FORTRAN IV Compiler (12K BG area)
24177-60002-B	RTE/DOS FORTRAN IV Compiler (12K BG area)
24248-60001-B	RTE/DOS Relocatable Library-Floating Point
29013-60001-C	7900 Moving Head Disc Driver DVR31
92001-12001-B	RTE-II Assembler
92001-16002-B	Relocating Loader
92001-16003-B	Multi-Terminal Monitor
92001-16004-C	Power Fail (DVP43)
92001-16005-B	RTE-II System Library
92001-16012-C	RTE-II Core Resident System
92001-16013-B	M.H. Generator
92001-16014-B	Auto Restart
92002-16010-B	Interactive Editor EDITR
92002-12001-C	Batch-Spool Monitor, D.RTR and FMGR
92002-12002-C	Batch-Spool Monitor, Spool
92002-16006-C	Batch-Spool Monitor Library

RTE DRIVERS

DVRXX NUMBER	PART NUMBER/ REVISION CODE	DESCRIPTION	MANUAL PART NUMBER
00	29029-60001-C	Multiple-Device System Control	29029-95001
11	29030-60001-B	HP 2892A Card Reader	09600-93010
12	29028-60001-B	HP 2610A/2614A Line Printer	–
12	29028-60002-A	HP 2767A Line Printer	–
12	92200-16001-A	HP 2607A Line Printer	92200-93001
15	09601-16021-A	HP 7261A Mark Sense Card Reader	09601-93014
23	92202-16001-A	HP 7970 Series 9-Track Magnetic Tape Units	92202-93001
24	25117-60499-D	HP 7970 Series 7-Track Magnetic Tape Units	25117-93003
31	29013-60001-C	HP 7900 Moving Head Disc Controller	–
40	29100-60041-A	HP 12604B Data Source Interface	29100-93001
40	20295-60001-A	HP 12604B Data Source Interface	12604-93002
54	25117-60551-A	HP 12556B 40-Bit Output Register	25117-93001
56	20297-60001-D	HP 2310/2311 Subsystem	02310-90004/ 02311-90001
57	20396-60001-A	HP 12564A 10-Bit A-to-D Card	12564-90041
61	14907-60001-D	HP 6940A/6941A Multiprogrammer	29100-93003
62	29009-60001-C	HP 2313B Analog-Digital Interface Subsystem	29009-93001

RTE DRIVERS (Cont'd)

DVRXX NUMBER	PART NUMBER/ REVISION CODE	DESCRIPTION	MANUAL PART NUMBER
65	29001-60001-D	HP 12771A Computer Serial Interface	12665-93001
65	29001-60003-C	Computer Serial Interface (Distributed Systems Driver)	91700-93001
66	29003-60001-A	HP 12770A Coupler Serial Interface	29003-93003
66	29003-60001-A	HP 12772A Coupler Modem Interface	29003-93001
70	25117-60414-B	HP 6129/6130/6131 Series Digital Voltage Sources	25117-93005
74	29000-60001-A	HP 2321A Low-Speed Data Acquisition Subsystem	02321-93001
76	20236-60001-A	HP 2320A Low-Speed Data Acquisition Subsystem	02320-93002
77	20235-60001-A	HP 2323A Low-Speed Data Acquisition Subsystem	02323-93001

RTE SUBROUTINES

PART NUMBER/ REVISION CODE	DESCRIPTION	MANUAL PART NUMBER
09601-16001-A	HP 12551B/12554A/12597A/12566B/12930A General Purpose Registers	09601-93005
09601-16009-A	HP 5326A-H18 Counter	09601-93009
09601-16010-A	HP 5327A/B-H48 Counter	09601-93007
09601-16011-A	RTE-B (BASIC Subroutine to HP 12604B Data Source Interface	09601-93013
09601-16012-A	HP 2320A DVM/Scanner Subsystem	09601-93011
09601-16014-A	HP 3480B/3484A Digital Multifunction Meter Subsystem	09601-93016
09601-16020-A	HP 12555B Digital-to-Analog Converter	09601-93017
09601-16022-A	HP 12556B 40-Bit Output Register	09601-93015
09601-16024-A	HP 2323A DVM Scanner Subsystem	09601-93018
20288-60001-A	Conversion Routine, BCD to Floating Point	02323-90001
29011-60001-E	HP 2313 Interface Main Module R2313	29009-93001
29011-60002-A	HP 2313 Interface Pacer Module P2313	29009-93001
29011-60003-A	HP 2313 Interface HP 2930 Module R2930	29009-93001
29011-60004-A	HP 2313 Interface Dual DAC Module D2313	29009-93001
29021-60001-A	FORTRAN/ALGOL Interface Subroutine DLK65 for RTE Driver DVR65	12665-93001
29100-16001-A	HP 6129/6130/6131 Digital Voltage Source Subsystem	29100-93005
29100-16003-A	Device Subroutine Library	29100-93007
29100-60044-A	HP 5326A/5327A Universal Timer/Counter, and HP 5326B/5327B Universal Timer/Counter/DVM Subsystems (FORTRAN only)	91064-93007

documentation

The following tables list all currently available HP 9600/9700 software manuals. This list supersedes the previous list in the **Communicator.** Copies of manuals and update packages can be obtained from your local Sales and Service Office. The address and telephone number of the office nearest to you are listed in the back of all reference manuals.

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manuals you need on the Corporate Parts Center form supplied at the back of the **Communicator.** If you require an update package only send your request to:

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

9600/9700 SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02005-90001	Real-Time Executive Software System	$12.00	10/01/71	
02320-93002	RTE System Driver DVR76 for HP 2320A Low Speed Data Acquisition Subsystem Programming and Operating Manual	1.00	08/03/74	
02321-93001	RTE System Driver DVR74 for HP 2321A Low Speed Data Acquisition Subsystem Programming and Operating Manual	1.00	08/30/74	
02891-90014	RTE 2891A Card Reader Driver (DVR11)		04/ /71	
09600-93010	RTE System DVR11 for HP 2892A Card Reader Programming and Operating Manual	1.00	08/03/74	
09601-93007	RTE Device Subroutine for HP 5327A/B-H48 Counter	2.50	12/ /74	
09601-93009	RTE Device Subroutine for HP 5326A-H18 Counter	2.50	12/ /74	
09601-93014	RTE System Driver DVR15 Mark Sense Card Reader Programming and Operating Manual	1.00	08/30/74	
09611-90009	9611A Operating 406 Industrial Measurement and Control System		04/17/75	
12665-93001	RTE System Driver DVR65 for HP 12771A Computer Serial Interface Kit	1.00	08/30/74	
12989-99001	RTE System Driver DVA 15 for Card Reader Punch Subsystem 2894	1.00	01/25/75	
25117-93003	RTE System Driver DVR24 for HP 7970 Series Digital Magnetic Tape Unit	1.00	08/30/74	
29003-93001	RTE System Driver DVR66 for HP 12772A Coupler Modem Interface Kit Programming and Operating Manual	1.00	08/30/74	
29003-93003	RTE System Driver DVR66 for HP 12770A Coupler Serial Interface Kit Programming and Operating Manual	1.00	08/30/74	
29009-93001	RTE System Driver DVR62 for HP 23138 Subsystem	2.50	08/30/74	
29013-90001	DVR31 RTE Moving Head Driver	10.00	02/01/73	
29014-90001	Moving Head Real-Time System Generator		04/01/72	
29015-90001	Fixed Head Real-Time System Generator			
29016-90002	RTE Scheduler		09/01/72	
29016-90003	Real-Time Input/Output Control		09/01/72	
29022-90001	Real-Time Relocating Loader	10.00	06/01/73	
29028-95001	RTE HP 2610A/2614A Line Printer Driver	1.50	10/31/72	
29029-91001	Real-Time Executive Multiple-Device System Control Device (DVR00) Program Listing	10.00	09/ /72	
29029-95001	Real-Time Executive System Driver DVR00 for Multiple Device System Control Small Programs Manual	1.00	10/10/72	03/ /75
29033-98000	Real-Time Executive File Manager System	10.00	03/01/73	
29100-93001	RTE System Driver DVR40 (29100-60041) for HP 12604B Data Source Interface Programming and Operating Manual	1.00	08/30/74	
29100-93003	RTE System Driver DVR61 for HP 6940A, 6941A Bidirectional Multiprogrammer Programming and Operating Manual	3.00	08/30/74	
29101-93001	RTE Core-Based Software System Users Manual	5.00	08/01/73	
91060-93005	RTE Driver for X-Y Display Storage Subsystem (HP Model 1331C-016) Programming and Operating Manual	1.00	08/15/74	

9600/9700 SYSTEM MANUALS (Cont'd)

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
91062-93003	Real-Time Executive System Driver for DVM/Scanner Subsystem	9.00	08/01/74	
92001-93001	Real-Time Executive II Software System	10.00	01/ /75	
92002-93001	RTE Batch-Spool Monitor Programming and Operating Manual	10.00	02/ /75	05/ /75
92200-93001	RTE System Driver DVR12 for HP 2607A Line Printer Programming and Operating Manual	1.00	08/30/74	
92200-93005	Real-Time Executive Operating System Drivers and Device Subroutine Manual	5.00	11/15/74	
92202-93001	RTE System Driver DVR23 for HP 7970 Series Digital Mag Tape Units Programming and Operating Manual	1.00	08/30/74	
93005-93005	Thermal Line Printer Subsystem for Driver DVR00 (RTE)	2.50	12/20/74	
93513-90002	RTE System Driver DVA 76-DVR40 for 2801 Quartz Thermometer Subsystem	1.50	04/30/75	

SOFTWARE INPUT/OUTPUT SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02100-90072	HP 2605A Console Printer Driver	$ 1.00	3/72	
02116-91760	Teleprinter Driver (LP Compatible) Manual	1.00	8/73	
02762-90002	HP 2762A Terminal Printer Driver	1.00	5/73	
02892-90003	HP 2892A Card Reader Driver	1.50	6/72	
12602-90022	Mark Sense Card Reader Drivers	1.00	6/70	
12653-90004	HP 2767 Line Printer Driver	1.00	9/70	1/73
12845-90005	HP 2610A/2614A Line Printer Driver	1.00	2/74	
12987-90006	HP 2607 Line Printer Driver	5.00	11/73	
13022-90010	HP 7970 Magnetic Tape Unit Driver	1.00	2/72	
13029-90010	Magnetic Tape Driver (7-Track)	1.00	2/72	
5950-9276	SIO Drum-Disc	1.00	2/70	
5951-1374	Software Input/Output System Configuration	1.00	7/74	
5951-1390	Subsystem Operation	2.00	10/74	

BASIC CONTROL SYSTEM MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
02100-90073	HP 2605A Console Printer Driver	$ 1.00	3/72	6/72
02116-9017	Basic Control System Manual	8.50	12/71	
02762-90003	HP 2762A Terminal Printer Driver	1.00	5/73	
02892-90004	HP 2892A Card Reader Driver	1.50	6/72	
12602-90021	Mark Sense Drivers	1.00	6/70	
12653-90005	HP 2767 Line Printer Driver	1.00	10/70	
12845-90004	HP 2610A/2614A Line Printer Driver	1.00	6/72	
12987-90008	HP 2607 Line Printer Driver	5.00	12/73	
13023-90010	HP 7970 Magnetic Tape Unit Driver	1.00	5/74	
13026-90010	Magnetic Tape Driver (7-Track without DMA)	1.00	5/71	6/72
13027-90010	Magnetic Tape Driver (7-Track with DMA)	1.00	5/71	6/72
5951-1388	Generating HP Basic	1.00	5/71	
5951-1391	Basic Control System	1.50	10/74	

training schedule

The schedule for training courses related to HP 9600/9700 systems is presented below. Included are courses scheduled for the next four months (September through December).

COURSE NUMBER	COURSE TITLE	COURSE LENGTH	DATES	LOCATION
SOFTWARE COURSES				
22965A	Real Time Measurement & Control	10 Days	Sept 8-19	Cupertino
			Sept 8-19	Rockville
			Oct 6-17	Cupertino
			Oct 6-17	Rockville
			Nov 3-14	Cupertino
			Nov 3-14	Rockville
			Dec 1-12	Cupertino
			Dec 8-19	Rockville
22966A	RTE Operating System	5 Days		
	This course is separately available as a one week course. It is also the first week of the two week 22965A Real Time Measurement & Control course listed above.			
22967A	Batch Spool Monitor	3 Days		
	This course is separately available as a 3 day course. It is also the first 3 days of the second week of the 22965A Real Time Measurement & Control course listed above.			
22968A	Measurement & Control Subsystem	2 Days		
	This course is separately available as a 2 day course. It is also the last 2 days of the second week of the 22965A Real Time Measurement and Control course listed above.			
22969A	Distributed Systems	5 Days	Sept 22-26	Cupertino
			Nov 17-21	Rockville
			Dec 15-19	Cupertino
22950A	2100 Minicomputer Assembler	5 Days	Sept 22-26	Rockville
			Sept 29—Oct 3	Cupertino
			Nov 17-21	Cupertino
			Dec 1-5	Rockville
22959A	21MX Minicomputer Assembler	5 Days	Sept 8-12	Rockville
			Oct 20-24	Cupertino
			Oct 20-24	Rockville
			Nov 17-21	Rockville
			Dec 8-12	Rockville
22974A	Minicomputers in Manufacturing	2 Days	Oct 16-17	Cupertino
HARDWARE COURSES				
22940A	2100A Operations & Maintenance	10 Days	Sept 8-19	Rockville
			Oct 6-17	Cupertino
			Nov 3-14	Rockville
			Dec 1-12	Cupertino
22941A	21MX Operations & Maintenance	5 Days	Sept 8-12	Cupertino
			Sept 29—Oct 3	Rockville
			Nov 17-21	Cupertino
			Dec 15-19	Rockville
22942A	7900A Operations & Maintenance	5 Days	Oct 20-24	Cupertino
22943A	7970B Operations & Maintenance	5 Days	Sept 15-19	Cupertino
22944A	7970E Operations & Maintenance	5 Days	Nov 3-7	Cupertino

HP Training Centers

Training is conducted in the U.S.A. at facilities in Cupertino, California and Rockville, Maryland.

Each Training Center is staffed with professional instructors. Courses are designed such that the student will receive both classroom instruction and practical, hands-on experience. By attending the courses in the recommended sequence for your particular HP system, the student will gain the most beneficial training available to meet the needs of your specific application.

Western Training Center

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

Eastern Training Center

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

Accommodations

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

Registration

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

Cancellations

In the event you find you cannot attend a particular class we would appreciate your prompt notification.

bulletins

NEW SEMINAR BEING OFFERED

"MINICOMPUTERS FOR MANUFACTURING PRODUCTION AND INVENTORY CONTROL"

The training schedule in this issue lists a new seminar entitled "Minicomputers in Manufacturing."

Why The Interest In Minicomputer Production and Inventory Control Application?

Applications of minicomputers for manufacturing production and inventory control are increasing at a very rapid rate. This is due to companies realizing that corporate profit can be increased by installing minicomputer systems that produce tangible benefits such as: reduced inventory investment, lower material costs and increased plant productivity.

About The Seminar.

This seminar addresses how today's minicomputer capabilities can effectively be applied to create systems for Engineering Data Control, Forecasting, Master Scheduling, Material Requirements Planning, Inventory Management, Shop Floor Control and Purchasing.

Desirable minicomputer capabilities discussed will be, Data Base Systems and Data Base Query, Data Communications, on-line Program Development and Peripherals sub-systems.

The course will discuss information screens or reports for each application area as well as how to design the files and data base necessary for system implementation.

When Will The Seminars Be Given?

If you are interested in attending the first seminar to be given August 28-29 in Cupertino, contact your local HP representative. The next seminar will be presented October 16-17 also in Cupertino.

Chuck Brewer
HP Cupertino

UPDATING YOUR MANUAL

Are you sure that your manual is up-to-date?

Change pages to existing manuals can be obtained from the Software Distribution Center at no extra charge. To obtain your copies of the changes, simply fill in your name and address and the name and part number of the manual update you require, and mail as directed.

REQUEST FOR UPDATE PACKAGE

Name of Manual ______________________________

Part Number of Manual ______________________________

Your Name ______________________________

Your Address ______________________________

City, State and zip code ______________________________

Area code and telephone number ______________________________

MAIL TO: Software/Publications Distribution
Hewlett-Packard
Data Systems Division
11000 Wolfe Road
Cupertino, CA 95014

Ilene Birkwood
HP Cupertino

featuring - HP's new line-up of support

In a continuing effort to ensure quality support of HP's computer products, the field and factory support organizations have recently redefined their objectives and responsibilities.

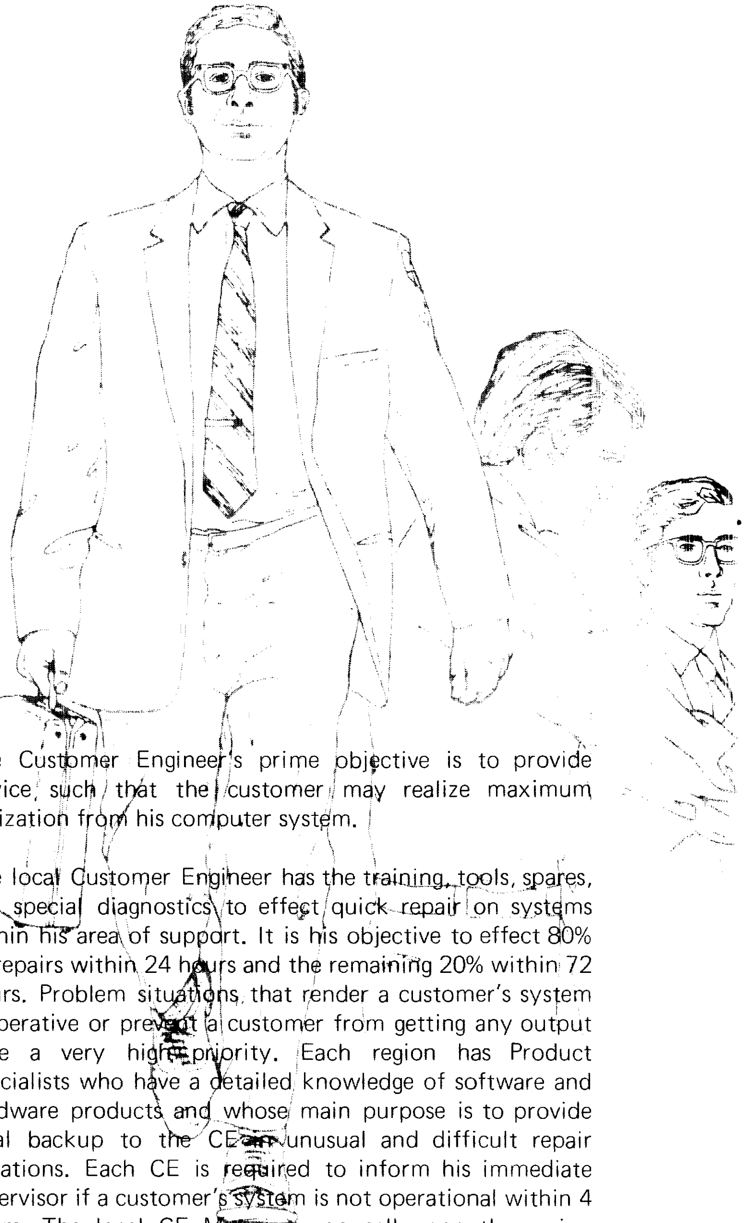
This issue of the Communicator brings you two articles which outline our support policies and describe how they affect you and your computer installation.

Wil Houde, Computer Systems Group Customer Engineering Manager describes the steps taken at the local level to keep your computer system operating at maximum efficiency.

Jerry Peterson, Customer Engineering Manager of Distributed Data Base Systems, explains the structure of the back-up support organization at the division level.

LOCAL CUSTOMER ENGINEERING SUPPORT FOR YOUR COMPUTER SYSTEM

It is the intention of Hewlett-Packard to provide its customers with high quality support consistent with fair price and quick response. To this endeavor there are over 500 Customer Engineers and Product Specialists devoted to providing after-sale support from 172 offices in 65 countries. Each local office consists of a group of product knowledgable Customer Engineers who have been trained in both hardware and software aspects of the systems sold and supported by HP.



The Customer Engineer's prime objective is to provide service, such that the customer may realize maximum utilization from his computer system.

The local Customer Engineer has the training, tools, spares, and special diagnostics to effect quick repair on systems within his area of support. It is his objective to effect 80% of repairs within 24 hours and the remaining 20% within 72 hours. Problem situations that render a customer's system inoperative or prevent a customer from getting any output have a very high priority. Each region has Product Specialists who have a detailed knowledge of software and hardware products and whose main purpose is to provide local backup to the CE on unusual and difficult repair situations. Each CE is required to inform his immediate supervisor if a customer's system is not operational within 4 hours. The local CE Managers can call upon the region resources and if a problem exists for more than 8 hours, the manufacturing division's support team is called in for assistance.

Each hardware repair or software modification is entered into a computerized system called ROPS (Repair Order Processing System) which keeps track of all repairs performed in the field. Historical failure information, parts stocking, preventive maintenance schedules, and service planning are all derived from information stored in ROPS. A dynamic parts usage report allows the field CE to continuously evaluate his parts usage assuring that he always has the proper parts necessary to get the job done.

An important aspect of the CE'S job is developing a good rapport with the customer, understanding the customer's application and becoming part of the customer's team as far as keeping the equipment operational. The CE has all of HP's resources at his disposal and it is up to him to make the proper judgment to keep the system operational.

In providing support of Hewlett-Packard software the Customer Engineer is responsible for making the proper judgments regarding what resources are required to solve the problem. It is the Customer Engineer's responsibility to categorize the software problem in one of the following categories:

- Software Failure
- Product Enhancement
- Documentation Clarification
- User Error

A **software failure** is defined as a design defect in a software product which may result in the creation of one or more of the following conditions during the course of ***normal usage*** as described in the reference manual:

- System will no longer accept operator commands.
- All program execution ceases and cannot be reinstated.
- I/O ceases for one or more peripherals and cannot be reinitiated without restarting.
- The system or the user's data base cannot be retrieved.
- Repetitive operator intervention is required to maintain normal system operation.
- Performance of the product, as defined by the data sheet, cannot be achieved.

The above problems should be repeatable and demonstrable to Hewlett-Packard and the customer.

A **product enhancement** may be defined as a change to the product to provide additional capability or to improve its performance. Quite often, a customer wants the software product to perform in a manner that was not contemplated during the design of the product. In order to render such performance, a product enhancement is required.

A **documentation clarification** may be defined as the necessary addition of information to, or refinement of, existing documentation in order to clarify a particular aspect of a software product's performance. A documentation clarification may also be defined as a typographical error, a statement which is made that is clearly not intended, or a statement which may be ambiguous.

When the Customer Engineer encounters a software failure, he will investigate it, verify it, and then report it to the responsible Product Support Engineer at the division which manufactured the product.

The responsible Product Support Engineer will then attempt to determine the cause and extent of the design defect. Once the cause and extent of the design defect has been determined, the Product Support Engineer will attempt to develop a temporary fix or a method of programming around the defect. If the attempt is successful, the temporary solution to the problem will be communicated to the Customer Engineer.

The Product Support Engineer will then document the failure and report it to the responsible Production Engineer in the Manufacturing Department. A copy is also sent to the Quality Assurance Department.

The responsible Production Engineer, working in conjunction with the product's design engineer, will then implement the necessary revisions to the software to correct the defect within a period of sixty days after it is reported.

When the revised version of the software product is available, it will be sent to the field for installation.

When the Customer Engineer encounters a problem which is in the category of a product enhancement, he will immediately advise the customer that any changes necessary to remedy the problem are in fact product enhancements.

In the case of a documentation clarification, the CE gathers all of the pertinent information and reports it to the responsible Product Support Engineer at the division which manufactured the product.

The Product Support Engineer will pass this information along to the responsible Production Engineer in the Manufacturing Department.

The Production Engineer, working in conjunction with the Technical Publications Group, will then implement the necessary revisions to the documentation to clarify the particular aspect of product performance or correct the error.

Providing the kind of satisfaction that HP customers deserve requires the teamwork of the field and factory organizations. It is our endeavor to continue to improve the kind of support required. We are open for suggestions and would appreciate comments from our customers as to whether or not we are meeting your expectations.

Wil Houde

CUSTOMER ENGINEERING SUPPORT AT HEWLETT-PACKARD DATA SYSTEMS DIVISION

Customer Engineering Support tasks at the division level can be broken into four broad categories:

- Customer Services
- On-Line Technical Support
- Product Specialization
- Documentation

The **Customer Services** major job is the development of user training courses that customers may purchase to gain knowledge on operating our HP Computer System and how to apply that knowledge to typical problems. These training courses are packaged and sent to our training centers world-wide assuring a consistently high level of course material wherever customer training is offered. Customer Services operates two U.S. training centers — one at the Data Systems Division in Cupertino, California, the other at Rockville, Maryland.

Another important job of Customer Services is writing, gathering and editing material for the **Computer Systems Communicator.** The objective here is to provide the customer not only with timely status information about software products, documentation, and training courses, but also useful applications information on how to use HP computer products.

Finally, it's the job of Customer Services to maintain the three contributed software libraries at Data Systems. These are 1) the HP3000 contributed library, 2) 2000/ACCESS BASIC library 3) 2100/21MX non-BASIC library. When a customer contributes a program to one of these libraries, Customer Services will verify operation of the program, put the documentation into proper format and physically add the program to the library.

The **On-Line Technical Support** groups are the factory back-up for your local Customer Engineer and Systems Engineer. The groups consist of Product Support Engineers expert in software and hardware and in knowledge of how to use or fix HP Computer products. This means your local S.E. or C.E. has one place to call for technical assistance whether it be a difficult applications problem, verification of a software bug, or help in isolating a stubborn hardware problem. Our goal is to resolve most field support inquiries in the Technical Support Group itself, but if that cannot be done for some reason, back-up is available in the Product Specialist area and ultimately the Computer Systems Lab.

The **Product Specialist** groups are the Support planners of the Division C.E. Group. When a new project gets underway in the Computer Systems Lab one or more Product Support Engineers from the Specialist Group is assigned to the project. Their responsibility is to develop a Product Support Plan for that product, to work closely with the lab to ensure that supportability of the product is taken into consideration in the design of the product, and to work closely with Product Management to make sure our marketing goals for the product can be supported. The Product Support Plan will include plans for training C.E.'s world-wide on product maintenance, training S.E.'s world-wide on the technical features and special applications of the product, C.E. "fix-it" documentation, customer "use-it" documentation, maintenance contract, pricing, and, if the product is hardware, service kit contents, kit distribution and estimated MTBF (Meantime between Failure) and MTTR (Meantime to Repair).

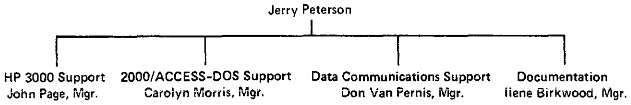
Once the plan is complete, the responsible product support engineer must see to the implementation, making sure that all the various groups involved are scheduled properly to complete their tasks in time for product release.

Last, but certainly not least, the **Documentation** sections of the C.E. groups have the dual tasks of producing C.E. "fix-it" documentation and customer "use-it" documentation. The objective of our C.E. documentation is to provide your local C.E. with the product information and diagnostic procedures necessary to quickly diagnose a product problem. Our user documentation, on the other hand, provides the user with a complete description of the product and is oriented towards helping the user become self-sufficient in using his system as quickly as possible.

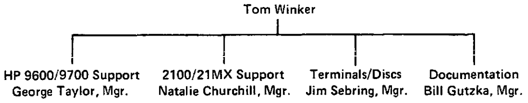
If all this sounds complex, you're right, it is. To organize this total support effort as efficiently as possible, we have recently reorganized the former Systems Engineering Group and Customer Engineering Groups at Data Systems into two product line oriented Customer Engineering Groups.

The product grouping was done along lines of major customer applications as follows:

C.E. SUPPORT – DISTRIBUTED DATA BASE SYSTEMS



C.E. SUPPORT – COMPONENTS AND INDUSTRIAL SYSTEMS



Providing administrative services to these two groups is a Support Operations Group headed by Ed Smith. We're doing our best to provide your local S.E. and C.E. with the best possible technical support. Part of this support is what you're reading right now – the Communicator. It's our prime vehicle to provide you with information to help you towards a successful installation.

Let us know how we're doing. We can always use feedback on how to improve our support. Write to the Communicator Editor or to any of us directly.

Jerry Peterson

software tips

This section of the Communicator contains timely how-to-do-it information contributed by HP customers and personnel. The information presented is pertinent to HP 3000 systems only.

You may find just the solution you've been looking for, or merely a better way to accomplish a particular task. If you've found a solution on your own, why not send it to the Communicator so that everyone can share it.

GETTING SPECIFIC ABOUT OUTPUT SPOOFLES

The article in Issue 1 of the Communicator on calculating the approximate size of output spoofles needs clarification. The article mentions that since records normally vary considerably in length, an average record length must be estimated in order to follow the calculations described.

The implication is that trailing blanks are truncated from each record but this was not stated specifically. Output spoofles *do* contain variable length records – trailing blanks are in fact suppressed.

Madeline A. Lombaerde
HP Cupertino

SPOOLING AND JOB MANAGEMENT NOTES

We will be publishing 3 separate articles related to Spooling and Job Management in this and the next two issues of the Communicator. The first article deals with User Facilities; the second is devoted to Console Commands and Device Operation; and the third article addresses Installation Management. The information is taken from a previously published System Analyst Note (3000.MPE.GENERAL-22, January 10, 1975).

Installment 1: User Facilities

1. JOB Submission

1.1 Sequential JOBs

Normally, every JOB in a sequence of JOBs is independent of the other. Such JOBs can be submitted and executed in any order. In certain exceptional cases, however, a sequence of JOBs is ordered such that a particular JOB must fully complete before the following JOB(s) are to be executed or even submitted. When such a JOB sequence is submitted on a non-spooled device, this ordering is implicitly effected, because every command record is executed when it is physically read. Spooling a job-accepting device can essentially nullify this ordering since: 1) JOBs are admitted (by the spooler) before preceding JOBs have executed; and 2) the concurrency of JOB execution is not necessarily limited to 1.

A JOB contains a grouping of requests which are presumably logically related in the indicated order. Every request pertaining to the JOB's chore is included; and no function not relating to the JOB is included. That's why JOBs are treated independently in MPE. When an "ordered" sequence of JOBs is to be submitted, the following can be considered:

1) Why are the JOBs all separate? The applications in which separate ordered JOBs must be submitted are rare. Users are encouraged to combine into one JOB all those functions necessary to implement the JOB's chore. Different JOBs are required only when a separate logon domain is necessary, such as account creation JOBs or to cause billing to another account, group or user. [Sometimes, users use separate, ordered JOBs so that the abortion of one step will not cause succeeding steps to be skipped; but :CONTINUE should be used for this.] The next two suggestions address those applications where an ordered JOB sequence is required.

2) Unspool device. This is obviously the simplest procedure. It makes the JOB sequence ordered because JOB execution is tied to physical reading.

3) "Spawn" JOBs with STREAM. A sequence of two or more disjoint, ordered JOBs can be converted into one JOB, in which every JOB is :STREAMed by its immediate predecessor. For example, the following two sequences of three JOBs are equivalent:

```
:JOB  J1         :JOB  J1
   :                :
:EOJ             :STREAM ,!
:JOB  J2         !JOB  J2
   :                :    (leading ":" replaced by "!"
:EOJ             !STREAM ,#
:JOB  J3         #JOB  J3
   :                :    (leading ":" replaced by "#"
:EOJ             #EOJ  J3
                 !EOD
                 !EOJ  J2
                 :EOD
                 :EOJ  J1
```

Note that the :STREAMed version is properly nested; i.e., the third JOB is :STREAMed by the second, not by the first–disjoint :STREAMed JOBs all originating from the outer (first) JOB would not work. This would not work if STREAM was not enabled, of course. Also, see Note 1.7 for special conventions necessary where nesting :STREAMs.

1.2 *:STREAM Disc Files*

When using the EDITOR to create :STREAM files, the user must save the text with the KEEP filename, UNN option. If it isn't, :STREAM will reject JOB/DATA commands with a syntax error (typically illegal parameter) because the sequence numbers are interpreted as part of the last parameters of the records.

1.3 *:STREAM from Cards*

When preparing :STREAM source from cards on the IBM 029 keypunch, remember that certain 029 and ASCII (Hollerith) characters are transposed. In particular, 029's "!" ***is not*** ASCII "!" (it's ASCII "] "). Place "!" in the :STREAM command (:STREAM,!), or use another character.

1.4 *":STREAMing Nothing"*

:STREAM performs a recognition function similar to that done for real devices. For example, the first action is to "flush" (ignore) the stream until a legitimate :JOB or :DATA command is found. No error message or job number will appear if no :JOB or :DATA command is found. This will be the case if the prompt character supplied (explicitly or implicitly) by the :STREAM command is different than :STREAM source, as may sometimes be the case, mistakenly, when using a disc file.

1.5 *Terminating Batch :STREAMs*

Although SESSIONs can terminate :STREAM with break, JOBs must signal end-of-file. The derfault :STREAM input is $STDIN, so any ":" will terminate the operation, not the STREAMed JOB's EOJ. For example:

```
:JOB
:STREAM ,!
!JOB
 .
 .
 .
!EOJ
:FILE L. . .
 .
 .
 .
```

will result in the :FILE command being "swallowed" by :STREAM, signaling end-of-file to it. :EOD should be used to terminate :STREAM in JOBs.

1.6 *:STREAM :JOB/:DATA Recognition – Read Length*

It is possible to extend a terminal input line by using LF. However, because :STREAM reads only the record width of the source file, this should not be used for long :JOB/:DATA commands; use "&" continuation.

1.7 *Nested :STREAMs*

When one :STREAMed JOB is introduced from within another :STREAMed JOB be sure they use different prompt characters. Consider, for example:

```
  :JOB OUTER
     :
     :
┌ :STREAM ,!
│ !JOB MIDDLE
│    :
│    :
│┌!STREAM ,#
││#JOB INNER
││   :
││   :
││#EOJ
│└!EOD
│ !COMMENT THIS IS "MIDDLE"
│ !EOJ
└ :EOD
  :COMMENT DONE "OUTER's" STREAM
     :
     :
```

Here "OUTER" initiates one :STREAM JOB ("MIDDLE"); "MIDDLE", in turn initiates "INNER" when ***"MIDDLE" executes.*** [This is a method to ensure JOB sequentiality, see 1.1.1.] If "#" hadn't been used for "INNER", then "OUTER" would wind up initiating ***both*** "MIDDLE" ***and*** "INNER". Note, also, the proper nesting of EODs.

2. Interpreting Job/Session Output

When JOB $STDLIST to line printers is examined, the following times are noted:

_ _HEADER - Timestamp, T_h
  :JOB & WELCOME info - Timestamp, T_j
_:EOJ - Elapsed Time, E
  - Timestamp, T_e
TRAILER - Timestamp, T_t

The following attempts to clear up any confusion regarding the interpretation and relationship of these times:

T_h - the (wall) time that the header is *printed*;
T_j - the (wall) time that the JOB acutally *began execution*;
E - the time between the job finishing execution and the time that the JOB was *introduced*, which means that it includes the time that the JOB was *waiting* to begin processing;
T_e - the (wall) time that the JOB *finishes execution*;
T_t - the (wall) time that the trailer is *printed.*

In analyzing these times, keep in mind that they can be off by ± one minute, due to roundoff. The following relationships can be derived from the definitions above:

$T_h < T_t$
$T_j < T_e <= T_t$
$E >= T_e - T_j$ (E includes waiting time)
$T_h = T_j < T_e = T_t$, when $STDLIST not spooled
$T_j < T_e <= T_h < T_t$, when $STDLIST is spooled

[The equal relationship, above, should be interpreted as "close to".]

2.2 *"(INCOMPLETE)" Trailers*

The message "(INCOMPLETE)" on a trailer indicates that the spooled output has been interrupted and does not appear in its entirety. This will occur if:

a. The console operator has explicitly interrupted it by command while it was being printed. It can be "deferred" for possible later printing or deleted.

b. A disc I/O error was detected while it was being printed.

c. The system crashed while the output was being created (before FCLOSing the file). The console operator should know the specific reason.

2.3 *No Trailer*

A trailer will not appear with an output line printer or card punch file if the system crashes during printing of the file; or, if spooled, a spoolee (device) error is detected.

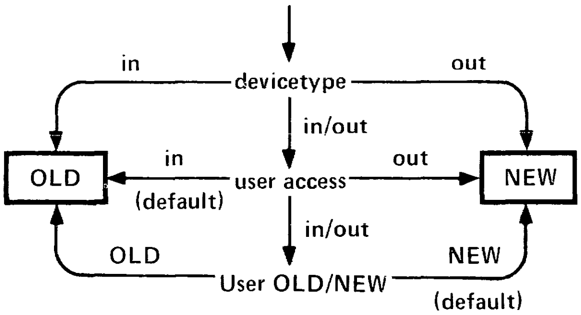
2.4 *Card Punch Headers/Trailers*

Files produced on spooled card punches are preceded by header cards and followed by trailer cards. These contain only descriptive information about the file which can be read by "interpreting" the cards.

3. Device Allocation

3.1 *Allocate Algorithm*

The first determination made when allocating a non-sharable device is whether the request is for an "OLD" or "NEW" devicefile:



OLD means to search for a pre-defined input file (e.g. :DATA,$STDIN). If an OLD file cannot be found, ***the console operator is queried.*** If a NEW request fails, the user's FOPEN is rejected. The console operator is asked to "locate" all mag tap requests.

Refer to the MPE manual for a discussion regarding the relevant names (job, file) when allocating a :DATA devicefile.

3.2 Allocating :DATA Terminals

:DATA devicefiles can only be allocated as OLD; see Note 3.1. When the target is a card reader, OLD is implied by the device type; i.e., input. But when the :DATA devicefile is a terminal (input/output device), the user specifications become relevant. In particular, he should not request output – only access; and if he's not requesting input access, he must specify OLD. Otherwise, he will get an FOPEN failure.

3.3 Device Assignment for Partially Spooled Classes

When a device class is configured such that some devices on it are spooled and some are not, NEW allocations will prefer ***available real*** devices to spooled devices.

Look for the article on Console Commands and Device Operation in the next issue of the **Communicator.**

PUTTING FORTRAN ROUTINES IN SLs

FORTRAN-written subprograms may be put into an MPE SL (Segmented Library) File provided certain restrictions are observed.

Q. Why would I want to do that?

A. Routines in an SL are shared by all programs that reference them. Only one copy of the routine is needed – this increases efficiency.

Q. Can Function Subprograms and Subroutines be used?

A. Yes. Both kinds. The system will check the parameters and function type in the usual way when you type the 'RUN' command for the program that references the routines.

Q. What are the restrictions?

A. (1) The routines must not use Global storage. In FORTRAN terms this means the routine cannot have COMMON statements or DATA statements of any kind.

(2) This may sound strange, but any READ/WRITE statements must not use a constant for the unit number. An integer variable is OK. For example, this is illegal:

```
      WRITE(6,10)
This is OK:
      I=6
      WRITE(I,10)
```

Don't forget to compile the calling program with a $CONTROL FILE=6 to create a flut entry for each number you use.

(3) You cannot use ACCEPT/DISPLAY statements. Use free-field READ/WRITES instead. (i.e., use asterisk instead of format reference in the READ/WRITE statement).

(4) You cannot use a FORMAT statement if it is referenced by a READ statement **and** it contains a Hollerith string.

This won't work:

```
READ(5,10)
10    FORMAT(6HABCDEF)
```

It is legal (but rare) in FORTRAN, but a routine containing it cannot be put into an SL.

Q. I have heard that I cannot do ***any*** I/O in such routines – is this true?

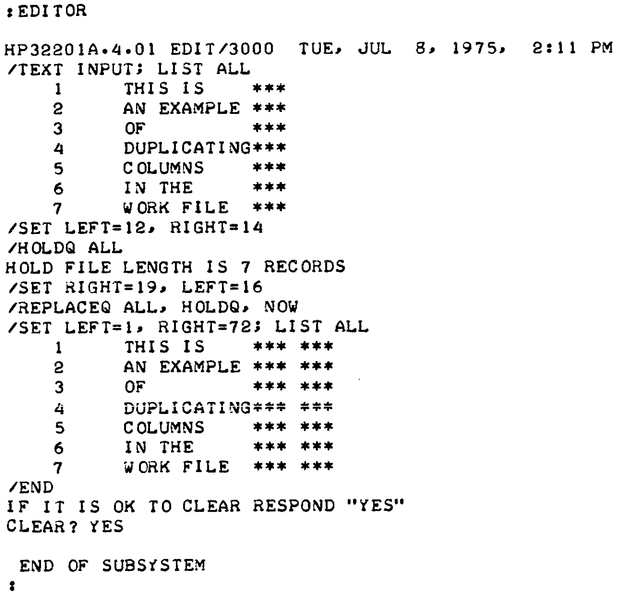
A. Absolutely not. Provided the above restrictions are observed you can use any READ/WRITE statement you like, including disc I/O.

Madeline Lombaerde
HP Cupertino

TIPS ON USING THE HP 3000 TEXT EDITOR

We will be giving you some tips from time to time on using the HP 3000 TEXT EDITOR.

The basis for this tip is from a presentation by Doug Mecham of Hughes Aircraft Co. at a regional HP 3000 Users Group meeting.



```
:EDITOR

HP32201A.4.01 EDIT/3000  TUE, JUL  8, 1975,  2:11 PM
/TEXT INPUT; LIST ALL
    1     THIS IS    ***
    2     AN EXAMPLE ***
    3     OF         ***
    4     DUPLICATING***
    5     COLUMNS    ***
    6     IN THE     ***
    7     WORK FILE  ***
/SET LEFT=12, RIGHT=14
/HOLDQ ALL
HOLD FILE LENGTH IS 7 RECORDS
/SET RIGHT=19, LEFT=16
/REPLACEQ ALL, HOLDQ, NOW
/SET LEFT=1, RIGHT=72; LIST ALL
    1     THIS IS    *** ***
    2     AN EXAMPLE *** ***
    3     OF         *** ***
    4     DUPLICATING*** ***
    5     COLUMNS    *** ***
    6     IN THE     *** ***
    7     WORK FILE  *** ***
/END
IF IT IS OK TO CLEAR RESPOND "YES"
CLEAR? YES

 END OF SUBSYSTEM
:
```

Send me any of your handy techniques using the HP 3000 Text Editor and will help you share them with other users.

Dick Sleght
HP Cupertino

EBCDIC TO ASCII CONVERSION

A new intrinsic was added with the release of MPE version C that might have gone unnoticed. It is called CTRANSLATE. It will translate EBCDIC to ASCII (or vice versa) with internally supplied tables. The intrinsic also allows for translation of other specific codes by user-supplied tables.

The CTRANSLATE intrinsic declaration is:

```
PROCEDURE CTRANSLATE (CODE, INSTRING,
                      OUTSTRING,
                      STRINGLENGTH,
                      TABLE);
  VALUE CODE, STRINGLENGTH;
  INTEGER CODE, STRINGLENGTH;
  BYTE ARRAY INSTRING, OUTSTRING, TABLE;
  OPTION VARIABLE, EXTERNAL;
```

The condition codes returned by the CTRANSLATE intrinsic:

CCE request granted.

CCL request denied because an error occurred.

CCG not returned.

The CTRANSLATE parameters are:

CODE An integer identifying a specific translation table to be used.

0 – use the user-supplied translation table given by the parameter TABLE.

1 – use the HP-supplied table for translating EBCDIC to ASCII. Those EBCDIC characters which have no ASCII equivalent will translate into a byte of zero.

2 – use the HP-supplied table for translating ASCII to EBCDIC. (ASCII parity bit is ignored).

INSTRING

the string of characters to be translated from one character set to another.

OUTSTRING

an optional parameter specifying a byte-array to which is returned the translated character string. If OUTSTRING is not specified, all translation will occur within INSTRING. The parameters INSTRING and OUTSTRING may specify the same array.

STRINGLENGTH

a positive integer specifying the length (in bytes) of INSTRING.

TABLE

a byte array whose ordering and contents define the transition process. This parameter is required only when CODE = 0.

This table is constructed such that, for each byte, a table entry is a byte in the target character set and is located at a byte position whose displacement from TABLE is the numeric value of the corresponding character in the source character set.

```
------------------------
   table(source) = target
------------------------
```

The length of TABLE may be as large as 256 bytes, but it needs to be only as large as the largest numeric value of any source byte in INSTRING.

Robert K. Strand
HP Cupertino

SEGMENT SIZES OF SUBSYSTEMS

Information concerning the number and sizes of segments is helpful to a user when allocating subsystems and resegmenting user programs. To determine the number of segments and segment sizes (code and data) the user should run PROGSTAT or PROGSIZE. These programs are available from the Contributed Library. PROGSTAT.PUB.SYS produces output formatted for the line printer. PROGSIZE.PUB.SYS produces output formatted for a terminal.

The output below is a sample from the running of RPOGSIZE.PUB.SYS. The program is terminated by responding :EOD to ENTER PROGRAM NAME:

```
RUN PROGSIZE.PUB.SYS



ENTER PROGRAM FILE NAME: EDITOR.PUB.SYS

*************************************************
*                                               *
*      PROGRAM NAME: EDITOR.PUB.SYS             *
*                                               *
*   SEGMENT SIZES    ******************         *
*                    *                * <--DL   *
*   SEG Ø    1796    * INITIAL SETTING *        *
*   SEG 1    1756    *       Ø        *         *
*   SEG 2    1212    *                *         *
*   SEG 3    2444    ******************         *
*   SEG 4    2896    *                * <--DB   *
*   SEG 5    1112    *     GLOBAL     *         *
*   SEG 6    2628    *    VARIABLES   *         *
*   SEG 7    2Ø36    *      956       *         *
*   SEG 8    648     *                *         *
*   SEG 9    2784    ******************         *
*   SEG 1Ø   2872    *                * <--Q    *
*   SEG 11   2Ø36    *     DYNAMIC    *         *
*   SEG 12   27ØØ    *      STACK     *         *
*   SEG 13   2172    *      3ØØØ      *         *
*   SEG 14   222Ø    *                *         *
*                    *                * <--Z    *
*  LARGEST:   2896   ******************         *
*  SMALLEST:  648                               *
*  AVERAGE:   2Ø87                              *
*                                               *
* TOTAL CODE: 31312  TOTAL DATA: 3956           *
* ---------------    ---------------            *
*************************************************
```

```
ENTER PROGRAM FILE NAME: FORTRAI\N.PUB.SYS

*************************************************
*                                               *
*      PROGRAM NAME: FORTRAN.PUB.SYS            *
*                                               *
*   SEGMENT SIZES    ******************         *
*                    *                * <--DL   *
*   SEG Ø    3116    * INITIAL SETTING *        *
*   SEG 1    28Ø8    *       Ø        *         *
*   SEG 2    1536    *                *         *
*   SEG 3    784     ******************         *
*   SEG 4    2ØØ8    *                * <--DB   *
*   SEG 5    1956    *     GLOBAL     *         *
*   SEG 6    19Ø8    *    VARIABLES   *         *
*   SEG 7    1692    *      13Ø9      *         *
*   SEG 8    1124    *                *         *
*   SEG 9    996     ******************         *
*   SEG 1Ø   1628    *                * <--Q    *
*   SEG 11   796     *     DYNAMIC    *         *
*   SEG 12   988     *      STACK     *         *
*   SEG 13   112Ø    *      25ØØ      *         *
*   SEG 14   19Ø8    *                *         *
*   SEG 15   142Ø    *                * <--Z    *
*   SEG 16   1572    ******************         *
*   SEG 17   1876                               *
*   SEG 18   192                                *
*   SEG 19   1756                               *
*   SEG 2Ø   528                                *
*                                               *
*  LARGEST:   3116                              *
*  SMALLEST:  192                               *
*  AVERAGE:   151Ø                              *
*                                               *
* TOTAL CODE: 31712  TOTAL DATA: 38Ø9           *
* ---------------    ---------------            *
*************************************************

ENTER PROGRAM FILE NAME:   :EOD
```

The following table gives a summary of the produced output for the remaining subsystems.

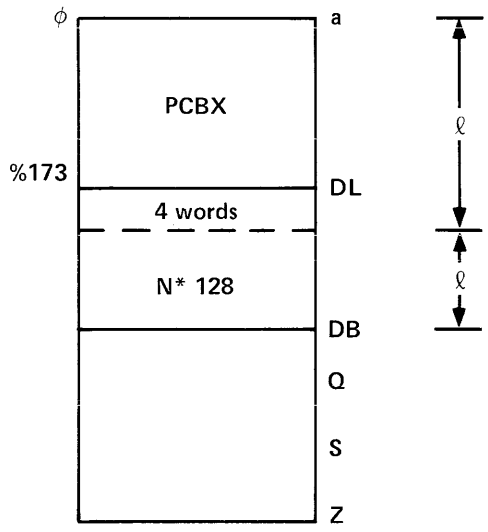
NAME	# of SEGMENTS	STATS	TOTAL CODE	TOTAL DATA
SPL.PUB.SYS	30	*L 3428 *S 248 *A 1239	37188	3974
SEGPROC.PUB.SYS	10	L 2064 S 492 A 1210	12100	1693
SEGDVR.PUB.SYS	1	L 856 S 856 A 856	856	1362
BASIC.PUB.SYS	26	L 3080 S 92 A 1602	41656	3560
BASICOMP.PUB.SYS	18	L 2944 S 536 A 1756	31608	1924
RPG.PUB.SYS	25	L 4040 S 520 A 2117	52936	3879
COBOL.PUB.SYS	28	L 3960 S 736 A 2294	64244	5684
COBOLB.PUB.SYS	31	L 4092 S 744 A 2406	74604	5685
DBSCHEMA.PUB.SYS	5	L 1612 S 856 A 1094	5472	5618
DBLOAD.PUB.SYS	1	L 1808 S 1808 A 1808	1808	5135

```
DBUNLOAN.PUB.SYS        1    L 2704    2704  3527
                             S 2704
                             A 2704

DBUTIL.PUB.SYS          1    L 1228    1228   803
                             S 1228
                             A 1228

DBSTORE.PUB.SYS         1    L 1648    1648   803
                             S 1648
                             A 1648

DBRESTOR.PUB.SYS        1    L 2428    2428   803
                             S 2428
                             A 2428

QUERY.PUB.SYS          12    L 3400   30000  5017
                             S 1960
                             A 2500

RJE.PUB.SYS             5    L 2300    5420  2077
                             S  452
                             A 1084

SORTB.PUB.SYS           1    L 2920    2920   801
                             S 2920
                             A 2920

STAR.PUB.SYS           17    L 2532   10892 16285
                             S   36
                             A  640

KA2100.PUB.SYS          5    L 2112    8308  2067
                             S  980
                             A 1661

XL2100.PUB.SYS          2    L 3408    6576  9125
                             S 3168
                             A 3288

FCOPY.PUB.SYS           5    L 2924    8816  1094
                             S  476
                             A 1763

SYSDUMP.PUB.SYS         5    L 3688   14356  4238
                             S 1868
                             A 2871

  *"L" Denotes Largest
   "S" Denotes Smallest
   "A" Denotes Average
```

CHANGE IN DLSIZE WITH MPE, VERSION C

Many people have inquired about a change in DLSIZE (DL to DB) with version C of MPE for the HP 3000. As you may know the permissible number of simultaneously opened files was increased. To accommodate the possible expansion beyond the initial 16 files and other changes within the PCBX of a user's stack, the size of the PCBX is now %174 under version C. This number is four words short of one sector of disc space and, since the distance from the beginning of the PCBX tó DB must be an integral number of sectors, it was decided to add those four words to the DL-area. Thus the statements in the manual that DL will be rounded up to·the next multiple of 128 words must be modified to read the next multiple of 128 words to which four is then added. Also the minimum DL-size possible is 132 words.

Robert K. Strand
HP Cupertino



$\ell = n * 128$ words where $n \geq 1$ with a default of 1

COMPLEX AND HYPERBOLIC FUNCTIONS NOW AVAILABLE

The HP 3000 BASIC User has some additional complex and hyperbolic arithmetic functions available. If you have purchased the BASIC compiler, one of the compiler segments contains these complex functions that can be called from FORTRAN and SPL. These functions extend the computational power of FORTRAN and SPL.

A description of these additional functions follows:

1. *Complex Hyperbolic Functions*

   Procedure CCOSH (Z); REAL ARRAY Z;
   Procedure CSINH (Z); REAL ARRAY Z;
   Procedure CTANH (Z); REAL ARRAY Z;

   [Calling sequence for SPL like that described on p. 2-36 of Compiler Library manual (HP 03000-90009) – user must stack 4 zeroes]

   From FORTRAN, type as complex and then assign result to complex variable (use as a complex function like CEXP, p. 2-21 Compiler Library manual)

2. *Double Precision Hyperbolic Functions*

   Long Procedure DCOSH (Y); LONG Y;
   Long Procedure DSINH (Y); LONG Y;
   Long Procedure DTANH (Y); LONG Y;

   From FORTRAN, type as DOUBLE PRECISION and assign result to double precision variable (use as a LONG function like DEXP, p. 2-20 Compiler Library manual)

3. *Additional Complex Function*

Procedure CTAN(Z); REAL ARRAY Z;

[User must stack 4 zeroes for the result]

Madeline A. Lombaerde
HP Cupertino

HP 2780/3780 EMULATOR ENHANCEMENTS

The Hewlett-Packard 2780/3780 Emulator is a sub-system of MPE/3000 that makes it possible for you to exchange data between your HP 3000 Computer System and a variety of remote processors over the public telephone (switched) network or private leased (non-switched) lines.

The 2780/3780 Emulator version B.01.00 contains many new enhancements which extend its capabilities and make it easier to use. The RJE command set is extended with new commands and additional parameters on the old commands to invoke the new features. A short discussion on each of the extensions follows.

#RJLINE...[;XEND]

The XEND parameter suppresses the RJE I/O Error: 3 printout. The condition causing the printout is not an error when communicating with a GE-High Speed Service and consequently the message printout was misleading.

#RJIN...[;MAXSIZE=nnn]

The MAXSIZE parameter extends the 80 byte limit logical record size sent by the emulator. The *nnn* part is positive for length in words and negative for bytes. Maximum *nnn* is approximately -500 or 250. This parameter is necessary to transfer USL files and program files between two 3000's.

#RJLIST...[;FORMSMSG=message]
[;OUTSIZE=*nnn*]
[;AUTOPAGE=$\left\{\begin{matrix}\text{YES}\\ \text{NO}\end{matrix}\right\}$]

The FORMSMSG parameter invokes the FILESYSTEM Forms Message when printing the received file. The message terminates with an optional period.

The OUTSIZE parameter suppresses the 80 byte truncation of transparent records received by the 3780 Emulator. *nnn* is length as described above in MAXSIZE parameter.

The AUTOPAGE parameter controls the carriage control when printing output received from a Host 360/370 computer.

YES means FWRITE carriage controls of %40, %60, or %304.

NO means FWRITE carriage controls of %201, %202, or %203.

#RJPUNCH...[;OUTSIZE=nnn]
Same use as #RJLIST

#RJOUT. . . [;OUTSIZE=nnn]
Same use as #RJLIST

#RJIO message

A new command applies to special situations in which a one line message is sent and a variable number of lines are received in response to the message. An example is a /*$DA sent to a Host 360 with HASP.

#RJIO message is equivalent to

```
#RJIN
message
#RJIOD
#RJOUT    ;WAIT=3
```

#message

This is equivalent to #RJIO message but the RJIO may be omitted for convenience. The message in this case must have any special character in column 1.

#RJINFO

A tombstone prints on $STDLIST and this may be helpful to the Customer Engineer in diagnosing problems.

#RJDEBUG

The capability to get into DEBUG through a command is another aid to Customer Engineers.

General Syntax Changes

- A Y or N can be used instead of YES or No in all commands.
- Messages such as the phone number and ID do not require quotes as delimeters. The quotes can be omitted.

3000 to 3000 file transfer capability has been considerably enhanced by the OUTSIZE parameter on #RJIN. For example, the RPG compiler has been successfully transmitted between two 3000's across the U.S. The following example shows the Emulator commands used to transfer a USL file.

SENDER

```
:JOB MANAGER.RJE;HIPRI;PRI=CS
:FILE USL=USLFILE,OLD
:RJE
#RJLINE 3780;LINECODE=ASCII
#RJIN *USL;XPARENT=YES;MAXSIZE=128
#RJEOD
#RJEND
:EOJ
```

RECEIVER

```
:JOB MANAGER.RJE;HIPRI;PRI=CS
:CONTINUE
:BUILD NEWUSL;CODE=USL
:FILE NEWUSL,OLD;NOCCTL
:RJE
#RJLINE 3780;LINECODE=ASCII
#RJOUT *NEWUSL;OUTSIZE=128
#RJEND
:EOJ
```

The following example shows the 2780/3780 Emulator commands used to transfer a DPAN memory dump tape.

SENDER

```
:JOB MANAGER.RJE;HIPRI;PRI=CS
:FILE DPANTAPE;REC=128,32,F;DEV=TAPE;BUF=1
:RJE
#RJLINE 3780;LINECODE=ASCII
#RJIN *DPANTAPE;XPARENT=YES;MAXSIZE=128
#RJEOD
#RJEND
:EOJ
```

RECEIVER

```
:JOB MANAGER.RJE;HIPRI;PRI=CS
:FILE  DPANTAPE;REC=128,32,F;DEV=TAPE;BUF=1;
  NOCCTL
:RJE
:RJLINE 3780;LINECODE=ASCII
#RJOUT *DPANTAPE;OUTSIZE=128
#RJEND
:EOJ
```

Jim Willits
HP Cupertino

bulletins

NEW USER MANUALS

FORTRAN/3000 USERS

An expanded FORTRAN/3000 Reference Manual is available. Examples are used extensively in the new edition, and demonstrate:

> Calls to MPE/3000 intrinsics from FORTRAN/3000 programs.
>
> How to create and maintain Relocatable Libraries (RL's) and Segmented Libraries (SL's), and how to call procedures in these libraries from FORTRAN/3000 programs.
>
> How to call the HP 3000 Compiler Library procedures FSET, FNUM, and UNITCONTROL.

The price of the new manual is $13.50. HP Part Number is 32102-90001.

Hal Goodwin
HP Cupertino

NEW SORT/3000 MANUAL AVAILABLE

Version B of the Sort and Merge software is reflected in a new SORT/3000 Reference Manual. The new manual contains examples which demonstrate how to run Sort and Merge as stand-alone programs in batch and interactive modes; and complete, operable programs showing how to call Sort and Merge procedures from SPL/3000 and FORTRAN/3000. The new manual Part Number is 32214-90001. The price is $6.50.

Hal Goodwin
HP Cupertino

EDIT/3000 USERS

A more comprehensive EDIT/3000 Reference Manual is available.

The original manual has been expanded to include:

Logging on to the system.

Using EDIT/3000 in batch jobs and interactive sessions.

Calling a FORTRAN/3000 procedure from EDIT/3000.

The HP Part Number is the same (03000-90012).

Hal Goodwin
HP Cupertino

RPG/3000 LISTING ANALYZER NOW AVAILABLE

If you are currently writing programs in the RPG/3000 language, HP's RPG Listing Analyzer can help you desk-check and debug them at the source level. The Analyzer is a 6" x 7", heavy-stock template that shows column numbers and field headings for all RPG specifications. When you place the Analyzer on your source listing and align it properly, you can tell immediately if each field contains the appropriate entry. Request the template by ordering HP Part No. 32104-90003. Price is 50 cents.

Dix McGuire
HP Cupertino

USING THE HP 3000

Subtitled *A Guide For The Terminal User,* this new manual fills a long-standing need for a simplified, hands-on introduction to the HP 3000. It instantly gets anyone on the machine, actually creating and running programs in the first sitting. Covers Editor, FORTRAN, COBOL, and BASIC. The format is specially designed for lap use while seated at a terminal, and 2-color printing is used throughout. Copies should be available about the time this issue is released. Single-copy price will be $7.50. HP Part Number is 03000-90121.

Ed Albert
HP Cupertino

WIN AN HP-65!

You can help us establish a formal, documented library of HP 3000 contributed software and receive a chance to win an HP-65 at the same time.

We are offering a prize of one HP-65 programmable calculator as an incentive for your help in gathering a library of fully documented programs. Our goals are to accumulate 100 useful contributions by March, 1976 and publish the abstracts in an indexed catalog by May, 1976.

Send your contributions according to the Contest Guidelines below. Submissions will be reviewed, and if accepted will become contest entries. Note that you may enter as many times as you wish. If you have six entries, you have six chances.

So, enter now. Simply cut out or copy the Documentation Form on the next page, arrange your submission as indicated on the form, and send in your :STORE tape with the completed form.

Contest Guidelines for HP 3000 Contributed Library Programs

A. Everyone except Hewlett-Packard personnel connected with the HP 3000 Users Group and the HP Contributed Software Center is eligible to enter the contest.

B. All entries must be received by March 1, 1976. Each accepted entry will have one chance to win through a drawing.

C. The drawing will take place March 15, 1976.

D. The prize is an HP-65.

E. All entries should be addressed to:

Hewlett-Packard
Contributed Software Center
11000 Wolfe Road
Cupertino, California 95014

F. Contest entries must be completely documented on an HP 3000 Contributed Software Documentation Form.

G. Programs must be written using HP supported programming language(s) and operate on an HP 3000 supported computer system.

H. To insure fairness, all submissions will be scanned for acceptance by a referee committee of HP personnel and HP 3000 Users Group members. Entries will be accepted if they can clearly and easily satisfy users' needs. Both HP and users are interested in programs with commercial, manufacturing, scientific and education applications.

I. You don't need to be present to win.

Brenda Mapp
HP Cupertino

HEWLETT-PACKARD CONTRIBUTED SOFTWARE CENTER

DOCUMENTATION FORM
HP 3000 CONTRIBUTED SOFTWARE

#______
(HP Use)

[ ] PROGRAM
[ ] SUBROUTINE } TITLE
[ ] PROCEDURE

ABSTRACT:

Key Words:__________, __________, __________, __________, __________

[ ] INITIAL SUBMISSION [ ] REVISION (PROGRAM #.__________)

REVISED TO: [ ] Correct Errors [ ] Clarify Documentation [ ] Other:

Submitted Files *(in LIB account):

Group:	DOC	JOB	SOURCE	PUB	Other______
FILENAME(s)	______	______	______	______	______
	______	______	______	______	______

Source Language:	[ ]	SPL	[ ]	FORTRAN	[ ]	RPG
(Main)	[ ]	COBOL	[ ]	BASIC	[ ]	__________
External Procedures:	[ ]	SPL	[ ]	FORTRAN	[ ]	RPG
	[ ]	COBOL	[ ]	BASIC	[ ]	__________

*DOC, JOB, and SOURCE files are required. When contributing software, please place your documentation into an unnumbered edit-file in the DOC group and the source in the SOURCE group. A job stream in the JOB group (:JOB MGR.LIB) should compile, install, and execute your submission as both a test case and an example of its usage. If your contribution is a complete program or an RL, it should reside in the PUB group. If you have contributed subprograms which can be placed in an SL, put the USL in the PUB group. Please try to use the same FILENAME for each of your files in the DOC, JOB, SOURCE, and PUB groups wherein practicable. Thank you for your consideration and contribution.

Required at Runtime: [ ] Compiler Library [ ] COBOL Library [ ] SORT/MERGE

[ ] RPG Library [ ] Plotter [ ] Line Printer [ ] IMAGE

[ ] Scientific Library [ ] TRACE [ ] Mag-tape [ ] Card Reader

[ ]________________ [ ]____________________ [ ]__________________ [ ]____________________

Other Software (Names and Sources):

Special Considerations, Restrictions, and/or Limitations:

References and Bibliography:

CONTRIBUTOR'S NAME:

ORGANIZATION:

ADDRESS:

TELEPHONE:

DISCLAIMER:

To the best of my knowledge, this contributed program is free of any proprietary information belonging to any person or organization. I am making this program information available to the HEWLETT-PACKARD CONTRIBUTED SOFTWARE CENTER. I hereby agree that HP may reproduce, publish, and use it, and authorize others to do so without obligations or liability of any kind.

______________________________ ______________________________

(Signature) (Date)

software updates

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

Software updates described in this issue relate to the following products:

MPE 32000C.00.06
MPE 32000C.00.07
HP 32100A SPL/3000
HP 32101A BASIC/3000
HP 32103A BASIC Compiler/3000
HP 32213 COBOL-A and HP 32213 COBOL-B
HP 32104A RPG/3000
HP 32215A IMAGE/3000
HP 32216A QUERY/3000
HP 32201A EDIT/3000
HP 30130B 2780/3780 EMULATOR 3000
HP 32204A STAR/3000
HP 32222A TRACE/3000
HP 32150 BUILDINT/3000
HP 32102A FORTRAN/3000
HP 32211C Compiler Library
HP OFFLN OFF-LINE DIAGNOSTICS
HP ONLN ON-LINE DIAGNOSTICS
MIT/Ship Date Schedule

Products described are available through your Customer Engineer, or can be ordered directly via Corporate Parts Center in Mountain View, California.

MPE 32000C.00.06 AND 32000C.00.07

This article corresponds to the June and July MIT tapes (date codes 1525 and 1531). Update information from these two tapes has been combined where possible. The two tapes comprise the official release of MPE 32000C.00.06 and 32000C.00.07.

The article is organized as follows:

I Modules Modified for MPE CO.06 & CO.07

II List of Problems Solved
  A. MPE CO.06
  B. MPE CO.07

III Enhancements to MPE
  A. MPE CO.06
    1. Spooling the Plotter
    2. CLOCK and CALENDAR intrinsics
    3. Change to the RECOVER program
  B. MPE CO.07
    1. Segmenter (use of USL created with BUILD)
    2. Spooling the Plotter - additional information
    3. Memory parity error detection
    4. Reload (skip over bad files)

IV Utilities
  A. MPE CO.06 Release of Supported Utilities (DISKEDIT,FREE,LISTDIR,PATCH, SLPATCH)
  B. MPE CO.06 Enhancements to Utilities

V Outstanding Problems (MPE CO-06 and CO.07)

VI Documentation Changes
  A. MPE CO.06
  B. MPE CO.07

VII Operational Notes MPE CO.06
  A. Recovering from a bad track on the disc
  B. Tape low condition for FWRITES to paper tape punch
  C. FORTRAN and HP 2894A users

The release of MPE CO.06 includes the release of another five (5) supported utilities. They are:

DISKEDIT.PUB.SYS
FREE.PUB.SYS
LISTDIR.PUB.SYS
PATCH.PUB.SYS
SLPATCH.PUB.SYS

These programs are now MPE supported and, as such, their source and maintenance files are available from Corporate Parts Center.

Part #32000 1X026: Source files

SDSKEDIT.HP32000.SUPPORT
SFREE.HP32000.SUPPORT
SLISTDIR.HP32000.SUPPORT
SPATCH.HP32000.SUPPORT
SSLPATCH.HP32000.SUPPORT

Part #32000 1X027: Maintenance files under HP32000.SUPPORT; namely:

USL Files	Maintenance Files	Job Files	Program Files
UDSKEDIT	MDSKEDIT	JDSKEDIT	PDSKEDIT
UFREE	MFREE	JFREE	PFREE
ULISTDIR	MLISTDIR	JLISTDIR	PLISTDIR
UPATCH	MPATCH	JPATCH	PPATCH
USLPATCH	MSLPATCH	JSLPATCH	PSLPATCH

I. Module Changes CO.06 and CO.07

		1	2	3	4	5	6	7
INITIAL	0	X		X	X			X
SYSDUMP	1	X	X	X			X	
SEGPROC	2	X	X				X	
SEGDVR	3							
DISPATCH	4			X			X	
LOAD	5		X					
MAPP	6					X		
UCOP	7	X						
DEVREC	8							
PROGEN	9	X						
ININ	10					X		X
EXIN	11	X	X	X		X	X	X
LOG	12	X						
IOPTRD0	13							
IOPTPN0	14						X	
IOPLOT0	15							
IOMDISK0	16			X				X
IOFDISK0	17			X				X
IOTAPE0	18				X			
IOLPRT0	19							
IOCDRD0	20		X				X	
IOCLTTY0	21							
IOTERM0	22							
IOCDPN0	23							
IOPRPN0	24					N	X	
IOREM0	25							
IOBSC0	26							
IOMDISK1	27	X		N				X
PFAIL	30			X	X	X		
FILESYS	50	X	X	X	X	X	X	X
COMM'INT	51	X		X			X	
STORE/RESTORE	52			X		X		
DIRC	53							
ALLOCATE	54		X		X			
DISKSPC	55	X						
MMCORER	56						X	
MMDISKR	57							
ABORTRAP	58						X	X
MESSAGE	59							X
CROUTINE	60			X	X			
IOUTILITY	61	X		X	X			X
TTYINT	62		X	X	X			X
PCREATE	63	X						
MORGUE	64			X				
PROCMAIL	65							
PINT	66						X	
DATASEG	67	X						
IOPM	68		X			X		
CHECKER	69							
UTILITY	70	X	X	X		X		
SEGUTIL	71	X		X				X
LOADER1	72		X	X				
RINS	73						X	
JOBTABLE	74	X						
DEBUG	75	X						
NURSERY	76			X				
SYSDPLY	77						X	
FIRMWARESIM	78	X						
SPOOLING	79			X	X			X
SPOOLCOMS	80	X					X	
MESSAGE CAT				X			X	X

MPE Supported Utilities

	5	6	7	8	9	
DISKEDIT		N				
DPAN	N					
FREE		N				
LISTDIR		N	X			
LISTEQ	N		X			
LISTLOG	N					
PATCH		N				
RECOVER	N	X				
SAEDIT	N		X			
SAVIOUR	N					
SLPATCH		N				

N: New Source Release
X: Changes (Maintenance File)

II A. List Of Problems Solved by MPE CO.06

a. There were cases where memory contention would cause the system to loop when a lower priority process (typically in the D Queue) was holding a chunk of memory frozen. This problem is fixed by having the dispatcher scan the lower queue when the situation arises.

b. Improper management of the USL directory was causing invalid entries in this directory to be generated when COPYing COBOL sub-programs from one USL to another. Corrected.

c. The SYSDUMP program aborted for lack of stack space under certain circumstances (increased of the RIN table and store operation, for example). This was due to MAXDATA not being large enough. To correct that problem, SYSDUMP was rePREPed with MAXDATA=20480, which should allow at least 1023 Global RINs.

d. The SETDUMP facility was modified to accept consistent syntax (see Documentation Changes) and to correct a problem which caused the ASCII option not to be recognized when the facility was run through an ABORT condition.

e. The system would fail in SYSDUMP if a dump of a full RIN table was asked. Fixed.

f. During a SYSDUMP, if a device was added which corresponded to a virtual device of the running system, the second word of the LPDT entry for that device was not reset, causing inconsistency in the allocation of that device.

g. An erroneous test in the Memory Manager would cause the swapping of stack whose size was exactly 16000 words to crash the system. Fixed.

h. The upper C and D Queues, although advertised as circular queues, were in fact treated as linear, causing erratic behavior when more than one process was scheduled in those subqueues.

i. The clock interrupt handler was changed to realize a compromise between two potential problems: appearance of negative times in the system and slow clock—whatever the load, the clock should be accurate within one minute a day.

j. The system was failing whenever a user would ask for more RINs (GETLOCRIN) than were available—the process was marked as having RINs allocated to it and the system would get a system failure #56 when that process was terminated. Fixed.

k. Access of an FCB by the File System was not protected against simultaneous access of the same FCB, which could potentially result in corruption of that FCB.

l. The EOT condition from Paper Tape Punch (tape low) is now handled by the system. The condition will cause a message to be emitted to the operator to mount a fresh reel of tape while the user program is suspended. A "YES" reply will cause CCE to be returned to caller. A "NO" reply will cause CCL to be returned. (See Operational notes for more details.)

Note:
This fix temporarily prohibits the use of Binary mode on paper tape punch. This will be corrected in a future release of MPE.

m. The problem listed under 'h.' in the previous release of MPE (CO.05) had not been fixed then but is fixed now. Repetitive calls to XCONTRAP to arm and disarm a C^Y procedure caused a bounds violation in the program under certain circumstances.

n. The Printer Reader Punch Subsystem (HP 30119A) driver has been changed—the device being a direct I/O device was locking out all throughput when spooled in at full speed. The change allows the throughput to stay up at the expense of reading not quite that fast from that device (120 card/minute maximum).

o. The card reader driver, IOCDRD0, has been enhanced to allow reading cards in the column binary or packed binary modes in addition to the ASCII mode. The driver is always set to the ASCII mode when a card devicefile is first opened but the mode may be changed via calls to the intrinsic FCONTROL with the ***controlcode*** parameter set to zero and the ***param*** parameter specifying the desired mode:

```
param = 0 column binary
      = 1 ASCII
      = 2 packed binary
```

The following special considerations must be observed when using either of the binary modes:

(1) The card reader must be accessed as a real device.
(2) NOBUF must be specified.
(3) End of file must be signaled by either the physical end of file on the card reader or by a card containing :EOF: in columns 1-5.

Finally, please note that the driver will no longer recognize a mode specified in a direct call to the MPE procedure ATTIO; the read mode can only be changed via FCONTROL.

II B. List Of Problems Solved By MPE CO.07

a. A trap procedure (control-Y or arithmetic) label which belongs to the GROUP SL, PUB SL, or SYS SL was mistakenly considered invalid.

b. The uncallable intrinsic PUTMSG could not be called in split stack mode without endangering the system.

c. Since not enough fill characters were sent to the Terminet at 1200 baud, the following solution has been adopted: 15 nulls (the maximum for the internal field) are sent after LF and 21 nulls are sent after CR. This will furnish 36 nulls for programs dealing with CRLF sequences. Programs using FWRITE or PRINT which send LFs followed by anything but a CR may suffer from characters being dropped.

d. When outputting a spoofle to a device, no test was done to determine whether the last record was output correctly before sending corresponding log record.

e. Origin of CST table was lost if last DRT was #30032 and the highest DRT was the last DRT + 1.

f. The default blocking factor for records with lengths of one (1) byte would result in blocking factor of 256 but would be internally stored as a blocking factor of zero (0) This could result in crashing the system during the file's creation. This has been fixed so that the default blocking factor is always computed as modulo 256.

g. The condition code resulting from a call to the PRINT intrinsic was being altered by an intervening call to EXCHANGEDB. This has been corrected in the procedure LOAD PROGRAM of the module LOADER1 so that the success of an error being printed will now be checked.

III A. Enhancements To MPE CO.06

1. Spooling the Plotter

Starting with this version of MPE, plotters can be spooled in output, which means that plotter logical devices become a valid argument to the command = SPOOL.

Consideration has to be given to the following points:

(1) The buffer size, when spooled, is limited to 256 characters which may affect the "zip" mode, and possibly the speed of the plotter, but not the end result of the drawing.

(2) If the system crashes while a spoofle is output to a plotter, the remaining part of the file will be useless at warmstart time—the spooling mechanism, as currently implemented, releases disc space as the spoofle is output. The spoofle will have to be deleted and the program run again. Consideration is given to retain the entire spoofle on the disc when spooling the plotter until the spoofle is closed.

(3) Two form messages are generated before the plot starts when using the PLOT routines of the library.

2. CLOCK and CALENDAR Intrinsics

Two new intrinsics are available that are a subset of the intrinsic CHRONOS. Both are in the SPLINTR file.

(1) *Time of Day*

A user can request the time of day from the system timer. The procedure CLOCK returns the time in a double word. The first word contains the hour (24-hour clock) and minute; the second word contains the second and tenth-of-a-second. The intrinsic is declared:

```
DOUBLE PROCEDURE    CLOCK;
OPTION EXTERNAL;
```

The format of the returned information is:

0 7	8 13
Hour	Minute
Second	Tenth-of-Second

The condition code is not changed by this intrinsic.

(2) *Date and Year*

A user can request the date and year from the system timer. The procedure CALENDAR returns this information in one word which contains the year (last two digits) and day-of-the-year. The intrinsic is declared:

```
LOGICAL PROCEDURE    CALENDAR;
OPTION EXTERNAL;
```

The format of the word returned is:

0 6	7 15
Last two digits of year	Day of Year

The condition code is not changed by this intrinsic.

3. Change to the RECOVER Program (CU.06)

Because the initial version (CU.05) of RECOVER failed re-creation of files if a duplicate file situation was encountered, an option has been added that will cause existing copies of files to be purged if the above condition is sensed.

The user will be prompted only once with the message:

```
WISH TO KEEP EXISTING COPIES OF
FILES? (Y/N)
```

Only a reply of "Y" will cause existing copies of files to be retained if a duplicate file condition exists, which makes it possible to run RECOVER after the system has been reloaded.

III. B. Enhancements to MPE CO.07

1. The SEGMENTER will now initialize an empty USL file before attemting to access it. Previous SEGMENTERS would abort due to UNEXPECTED EOF while attempting to access these uninitialized (created via BUILD command) USL files.

2. The routine that spools the plotter does not return the disc space (spoofle) until after the file is completely output (closed). This enables the system to allow a complete replot to take place after a warmstart (see Enhancements to MPE CO.06)

3. Memory Parity Error

   The Memory Parity Error in MPE has been modified to improve the ability of diagnosing a bad memory module.

   In case of Parity Error (Internal Interrupt #10), the system will:

   a. Halt with a System Failure #%130 message if the problem was a Memory Address Parity Error (parameter = %20000).

   b. Halt with a System Failure # %131 message if the problem was a system Parity Error (parameter = %40000).

   c. Follow different paths according to the environment; if the problems is a Data Parity Error (parameter = %10000), an attempt is made to detect the location of the bad address (XXXXX).

   - If the error occurred while running in user environment, the program is aborted (as before) and a message is output to the console. This message can be

     MEM DATA PARITY ERR.ADDR UNKNOWN.USER PROGRAM ABORTED

     or

     MEM DATA PARITY ERR.ADDR= XXXXX SYSTEM ATTEMPTS TO RESUME

     where XXXXX is the bad location address

   according to whether the system was successful in detecting the bad address.

   - If the error occurred in a critical area of the software, the system is halted with a System Failure Message # %132, possibly followed by the suspected bad location address if the attempt to determine it was successful.

   Notes:

   1) The address of the bad location should be kept whenever it appears; it should minimize the time to get around a memory problem.

   2) Since it is rather difficult to extensively test that mechanism, please report any malfunction that shows up.

   3) An examination of a core dump after a System Failure %132 would show segment %12 called from segment 0. This is necessary to avoid system halt when getting a Data Parity Error Interrupt on top of the same. Also, the dump would show the bad address in SYSB+239.

4. INITIAL now allows a user file to be skipped during a RELOAD if a Tape Parity Error is detected while reading the data portion of the file.

   The disc space is returned, the Account/Group disc space accounting is adjusted in the Directory, the file's identification is optionally displayed on the operator's console, and the file is skipped.

   The INITIAL console message

   INSUFFICIENT DISC SPACE – LIST FILES PURGED?

   has been replaced with a new message in order to accommodate the additional cause, TAPE PARITY ERROR, for failing to reload a user's file.

   The message now appears as

   FOLLOWING FILES PURGED
   BECAUSE OF:
   INSUFFICIENT DISC SPACE (1)
   TAPE PARITY ERROR (2)
   LIST?

   The numbers in parentheses will be appended to the file's name, when they are optionally listed, as an identification of the cause for the failure to reload the file. For example:

```
FOLLOWING FILES PURGED
BECAUSE OF:
INSUFFICIENT DISC SPACE (1)
TAPE PARITY ERROR (2)
LIST? YES
CROSSREF.PUB.SYS (2)
EDITOR.PUB.SYS (1)
LISTDIR.PUB.SYS (2)
```

In this example, CROSSREF.PUB.SYS and LISTDIR.PUB.SYS were purged because of cause (2): a Tape Parity Error occurred while reading those files. EDITOR.PUB.SYS was purged because of cause (1): there was not enough disc space available.

IV. Utilities

A. Release of Supported Utilities (MPE CO.06)

Another set of five (5) programs is released under this version of MPE. They can be found in the PUB.SYS group:

- DISKEDIT
- FREE
- LISTDIR
- PATCH
- SLPATCH

They perform essentially the same function as the programs of the same name, formerly known as "unsupported programs". It is highly recommended to purge the old version of those programs to keep only the versions hereby released.

DISKEDIT – On-line Disc Edit/Dump (CU.06)

This program displays and/or modifies a disc according to absolute sector addresses. It runs in privileged mode and requires system supervisor or system manager capability. Parameters are free format; octal numbers must be preceded by "%".

Commands:

BASE <absolute sector #>

Specifies the absolute sector number for the DUMP and MODIFY commands.

DEBUG

Calls system DEBUG program.

DISC <logical device #>

Specifies the disc.
Initially: 1

DUMP [<relative sector #>[,<# of sectors>]]

Displays the selected sector(s) on the LIST device. The starting sector address is determined by adding <relative sector #> and <absolute sector #> from BASE command.

Defaults:	relative sector	# = 0
	# of sectors	# = 1
Initially:	absolute sector	# = 0

EXIT

Terminates the program.

HELP

Displays a list of commands on $STDLIST.

LIST <logical device #>|<device class>

Specifies the device for the DUMP command listing.

Default:	$STDLIST
Initially:	$STDLIST

MODIFY <sector #, relative word address [,<# of words>]>

<sector #> – The absolute sector address (decimal or octal). Calculated by adding <sector #> to <absolute sector #> from BASE command.

<relative word address> – The word address of the first word to be modified, relative to word 0 of <sector #>. This may span sectors.

<# of words> – The number of words to modify. Default: 1.

This command prompts with the sector number being modified (octal), for each word to be modified:

SECTOR % <sector #> LDEV = % <logical device #>

<relative word address (octal)> : <contents of word (octal)> %

The valid responses are:

<new value> (CR)	–	new octal replacement, or
*	–	write the cur- value, or
/	–	abort MODIFY operation

The message "WRITTEN" is displayed after each sector is physically written. Disc contents are not altered before this message.

WIDTH

Specifies the ASCII DUMP command display be placed below or beside the octal dump. This command responds:

NARROW FORMAT?

A "Y" response places ASCII characters below the octal dump. Any other response places the ASCII characters to the right of the octal dump. The DUMP display writes records one character fewer than the record width of the LIST device. DUMP characters the record width and beyond are truncated.

Messages:

**PROGRAM REQUIRES SYS. SUP. OR SYS. MGR. CAPABILITY

Users of DISKEDIT must have system supervisor or system manager capability.

The following are error messages and include "**ERROR" in the line following the text of the message:

FWRITE ERR ON LIST <FCHECK #>

FWRITE returned not equal on the list device.

DISC <ATTIO #>

Attach I/O failed on the disc.

INVALID DISC ADDRESS

Address specified by BASE (DBINARY returned not equal) is invalid, or relative sector address plus absolute sector address in DUMP is less than zero, or sector address plus absolute sector address in MODIFY is less than zero.

DEVICE NOT DISC

GETDEVINFO indicates the device is not a disc.

UNABLE TO OPEN LIST DEVICE

FOPEN failed.

Caution:

> This program, by its ability to modify disc content, is extremly dangerous to the integrity of the system/files when misused. Its access should be restricted.

FREE (CU.06)

FREE is a utility program that allows a user to obtain a formatted detail listing and the total of a disc's Free Space Table entries. This information can be useful in observing the utilization of a disc's resources and is also useful in determining the degree of fragmentation of a disc's available space.

The program will prompt the user for the logical device number of the disc and will verify that the specified device is a disc. The listing, once it has started, can be discontinued by the use of the terminal's Control-Y function, and the user will again be prompted for another logical device number. The program can be terminated by responding with a carriage return to the logical device number question.

The formal designator for the program's output file is LIST, which defaults to $STDLIST.

```
:RUN FREE.PUB.SYS

DISC FREE SPACE LISTER (CU.06)

LOGICAL DEVICE #? 1
# OF ENTRIES = 2   TABLE SIZE = 16 SECTORS
MIN = %56   MAX = %37777
SECTOR (%)      LENGTH (%)
10137           1.
36456           1322

TOTSECT=        1323
```

```
LOGICAL DEVICE #? 3
DEVICE NOT DISC

LOGICAL DEVICE #? 2
# OF ENTRIES = 116   TABLE SIZE = 32 SECTORS
MIN = %76          MAX = %547277
SECTOR (%)         LENGTH (%)
1411               12
1623               2
1703               4
2104               1
2751               57
3042               36
3201               5
3225               6
3635               2
3726               3
4437               3
6173               4
6263               10
6335               4
Y^C                      (Terminate further input)
LOGICAL DEVICE #? CR   (To end program)
 END OF PROGRAM
:
```

LISTDIR

Introduction:

LISTDIR displays the attributes of specified accounts, groups, users, and files. The program provides information similar to that provided by the MPE commands :LISTACCT, :LISTGROUP, :LISTUSER, and :LISTF.

LISTDIR commands are read from $STDIN; prompts and error messages are directed to $STDLIST. Formatted output is written on the list file (formal designator: "OUT") which defaults to $STDLIST. This default can be overridden with a :FILE command by specifying ";PARM=1" on the :RUN command. Formatted output for an individual command can be directed to another file by specifying a list file on the LISTDIR command.

The execution of a command can be terminated by typing Control-Y. The user can enter system break mode (by typing the "break" key) only during the command phase of LISTDIR.

In general, a user may list the attributes of only his/her log-on account, group, and user name. As with the :LISTF command, the user may list the attributes of any file (although display of some attributes is restricted). Account and system managers may list other groups and accounts as well as display passwords, lockwords, and creator-ids. Additional security provisions are discussed later.

Command Format:

Commands consist of a command keyword and an optional parameter list. The command keyword must begin in the first column. Blanks may appear anywhere between syntactic elements (e.g., names, separators, keywords); note that a qualified name (e.g., "GROUP.ACCT") may not contain embedded blanks. The command keywords are:

```
LISTACCT
LISTGROUP
LISTUSER
LISTF
LISTSEC
HELP
EXIT
```

The syntax and effect of these commands are similar to the corresponding MPE commands. LISTF has the same effect as the MPE command ":LISTF fname,-1". LISTSEC displays the total file security for the specified file set. This includes account-, group-, and filelevel security as well as the "effective file security" for the log-on user. HELP provides a summary of the command syntax.

Commands:

The following is a description of the commands. The "listfile" is any valid file designator. If the file designator is to back-reference a :FILE command, an asterisk must precede the designator. If no asterisk is present, the file designator must be an existing job—temporary or permanent file. The syntax for these commands is given later.

The PASS parameter on most of the commands indicates that the password or the file lockword, creator-id, and disc address are to be displayed. If this parameter is not specified, or if the user does not have sufficient capability, "**" is displayed in place of the privileged information. The PASS parameter also allows display of privileged file codes.

LISTACCT

This command lists the attributes (e.g., capabilities, security) of a specified account or account set. If no account name is specified, the properties of the log-on account are listed. The account set may be an account name or the symbol @. Only a system manager may specify an account name other than the log-on account and may list the account password.

Form: LISTACCT [acctset] [,listfile] [;PASS]

LISTGROUP

This command lists the attributes (e.g., capabilities, security) of a specified group or group set. If no group name is specified, the properties of the log-on group are listed. The group set may be a group name optionally qualified by an account name, or the symbol @ optionally qualified by an account set (see LISTACCT). An account manager may specify any group or group set in the log-on account; a system manager may specify any group or group set. Only account and system managers may display group passwords.

Form: [groupset] [,listfile] [,PASS]

LISTUSER

This command lists the attributes (e.g., capabilities, home group) of a specified user or user set. If no user name is specified, the properties of the log-on user are listed. The user set may be a user name optionally qualified by an account name, or the symbol @ optionally qualified by an account set (see LISTACCT). When the specified user name is the log-on user, the job/session number, "home terminal" logical device number, and log-on group name are also listed. An account manager may specify any user name or user set in the log-on account; a system manager may specify any user name or user set. Only account and system managers may display user passwords.

Form: LISTUSER [userset] [,listfile] [;PASS]

LISTF

This command lists the attributes of a specified file or file set. The properties listed include the number of sectors used, creation date, disc unit number, file label address, and file security. For program files program capabilities and the number of segments are also listed.

The file set is a required parameter on this command. The file set may be a file name optionally qualified by a group and account name, or the symbol @ optionally qualified by a group set (see LISTGROUP). To prevent indiscriminate disclosure of group and account names, "standard users" may not specify the file sets "@.@", "@.@.acctname", and "@.@.@"; account managers may not specify the latter two file sets (where "acctname" is a name other than the log-on account).

The MAP and PASS parameters may appear in any order. MAP provides a list of the disc addresses for each extent. This is available only to the file creator and account and system managers.

Form: fileset [,listfile] [;PASS] [;MAP]

LISTSEC

This command displays the total file security for the specified file set. This includes the account-, group-, and file-level security as well as the "effective file security" for the log-on user. The "effective security" takes all levels of security into account and indicates the access available to the log-on user. The form and restrictions for this command are similar to those for LISTF.

Form: LISTSEC fileset [,listfile] [;PASS]

Syntax:

Commands:

```
LISTACCT     [ <aset> ]   [ ,<listfile> ]   [ ;PASS ]
LISTGROUP    [ <gset> ]   [ ,<listfile> ]   [ ;PASS ]
LISTUSER     [ <uset> ]   [ ,<listfile> ]   [ ;PASS ]
LISTSEC        <fset>     [ ,<listfile> ]   [ ;PASS ]
LISTF          <fset>     [ ,<listfile> ]   [ ;PASS ]   [ ;MAP ]
HELP
EXIT
```

Syntactic elements:

```
<aset>       : : =  <account name> : @
<gset>       : : =  <group name> [ . <account name> ]  : @ [ . <aset> ]
<uset>       : : =  <user name> [ . <account name> ]  : @ [ . <aset> ]
<fset>       : : =  <file name> [ . <group name> [ . <account name> ] ]  :
                ,   @ [ . <gset> ]
<listfile>   : : =  [ * : $ ]  <file name> [ / <lockword> ]
                    [ . <group name> [ . <account name> ]  ]
```

Security Restrictions:

A system manager is unrestricted.

An account manager may specify any group or user name in the log-on account.

A "standard user" (one who is neither a system manager nor an account manager) may specify only his/her log-on account, group, and user name.

Passwords, lockwords, creator-ids, file label addresses, and privileged file codes are displayed only when PASS is specified and only for qualified users (see below).

Account passwords may be listed only by a system manager.

Group and user passwords may be listed only by an account manager or a system manager.

File lockwords and creator-ids may be listed only by an account manager or a system manager.

File disc addresses and extent maps may be displayed by the creator of the file, an account manager, or a system manager.

Privileged file codes may be displayed by an account or system manager or by the creator of the file who has the privileged mode capability.

The file sets "@.@" and "@.@.acctname" may be specified only by an account manager (where "acctname" must be the log-on account) or by a system manager.

The file set "@.@.@" may be specified only by a system manager.

These restrictions are derived from MPE to prevent indiscriminate disclosure of group and account names and passwords as well as to prevent non-responsible users from tying up system resources.

PATCH (CO.06) – Program Patcher

This program can be used to display or modify the content of a program file (without recompilation) by directly accessing the object code. This may be the way to modify a program if the user is willing to examine his problem at instruction level and if the patch is simple and can be inserted in the sequence of machine instructions already existing.

(1) Running the program

After the output of the title, the program file name is prompted for:

```
FILE ? PROGNAME
```

where PROGNAME is entered by the user.

All numbers and addresses are in octal.

The prompt is "?". The user may then give one of the following directives:

D,segment number, address [,number of locations]
M,segment number, address [,number of locations]

in order to dump or modify a code segment. If changes are to be made to the global area of the initial stack, use:

```
M
  , D, address [,number of locations]
D
```

As shown above, the contents of the specified addresses are displayed. If number of locations is not specified, the default is 1.

Terminates with any character except M or D.

(2) Errors

For each kind of error, the program will abort with a specific number (QUIT).

QUIT	# 1	Program file could not be opened
	# 2	FREAD error on program file
	# 3	FCLOSE failure
	# 4	FREADIRECT failure
	# 5	FREADIRECT failure in a modify operation
	# 6	FWRITEDIRECT failure in a modify operation

(3) Example

```
: RUN PATCH

PROGRAM PATCHER CU.06
  FILE=?SLPATCH
?D,0,0,1
  140001
?D,0,0,5
  140001
  041001
  025026
  000600
  031020
?M,0,0,1
  140001,140002
?M,0,0,3
  140002,140001
  041001,041001
  025026,025026
? CR                To terminate program use a cr
                    or any character other than M or D
END OF PROGRAM
:
```

SLPATCH (CU.06) – Segmented Library Patcher

This program may be used to display and/or modify the content of an SL file.

(1) Running the program

All numbers and addresses are in octal. Upon request, the user gives the SL file name.

The file name for the system SL is SL.PUB.SYS.

Thereafter, the response to the prompt "?" is:

```
                 M
[Segment Name,]    , Segment Displacement [,no.
                 D
of words]
```

Segment Name is needed for the first response and whenever you want to change segments.

D is to display
M is to modify
Default for number of words is 1 when not specified.

To exit, enter EXIT followed by carriage return.

The following is an example:

```
: RUN SLPATCH

SL FILE PATCHER CU.06
SL FILE? SL.PUB.SYS
?SDMCOMM,D,0,5
025001
051404
041605
021040
005700
?D,0,5
025001
051404
041605
021040
005700
```

```
?M.0.1
025001 – 025002
?M.0
025002 – 025001
?
?EXIT

END OF PROGRAM
:
```

Caution:

This program is a privileged mode program, which is made necessary by the requirement that it should be able to access the SL.PUB.SYS even though it is being used.

The capabilities of this program make the system very vulnerable to its misuse, and for that reason, its access should be restricted.

(2) Errors

The program recognizes two kinds of fatal errors and quits in either case.

QUIT # 0 XXX IO ERROR *** *or* *** END OF FILE***
# 1 File Error = XX
where XX is the FCHECK number.

Non-fatal errors include:

```
ILLEGAL COMMAND
SEGMENT NOT SPECIFIED
ILLEGAL SEGMENT NAME
INVALID SL FILE
ILLEGAL RANGE
ILLEGAL NUMBER
```

B. Enhancement To Utilities (MPE CO.07)

LISTEQ

The utility program LISTEQ has been modified to disallow file equations for its output file LIST. File equations may be allowed by passing a parameter of 1 through the :RUN command. For example:

```
: FILE LIST=$NULL
: RUN LISTEQ                ←    The file equation
                                 is ignored and the
LISTEQ          (CU.07)          output is written
                                 to $STDLIST.
***NO TEMP FILES

***FILE EQUATIONS

: FILE LIST=$NULL
END OF PROGRAM
: RUN LISTEQ;PARM=1         ←    The file equation
                                 is used and the
END OF PROGRAM                   output is written
:                                to $NULL
```

LISTDIR

The following is a list of bugs fixed and enhancements.

(1) Creator of a file can now display creator ID.

(2) Hitting control-Y to abort a LIST-command no longer leaves buffer dirty.

(3) "CONTINUE?" query in HELP command did not accept lower case "Y" and "N".

(4) A list-file may now be specified on HELP command for off-line listing.

(5) "CONTINUE?" query is made only if LIST and $STDIN devices are interactive (e.g., a terminal).

(6) Trailing blanks are not printed.

SAEDIT

The stand-alone disc edit utility, SAEDIT, has been enhanced with two new commands, MTAPE and EXIT. The EXIT command terminates the program by halting the system. The MTAPE command allows dumping data from disc to tape or loading data from tape to disc. The primary purpose of this command is to allow preserving the entire contents of a disc while diagnostics are being run, but the command also allows dumping or loading only a portion of the disc. After recognizing the MTAPE command, the program will ask if you wish to dump or load. It will then ask for the number of sectors. If a number is entered, the program will dump (load) the specified number of sectors beginning at the base address. If a carriage return is entered, the entire disc will be dumped (loaded) regardless of the base address. When the dump is completed, a completion message is issued. The dump may be terminated early by setting switch 0.

V. Outstanding Problems in MPE (MPE CO.06 and MPE CO.07)

The outstanding problems in MPE CO.06 and CO.07 are the same with the following exception:

If the PLABEL passed to the procedures which arm traps in not part of the program file, but instead belongs to the library, an error is unduly returned to the caller.

(This outstanding problem in MPE CO.06 has been fixed in MPE CO.07; see Problems Solved by MPE CO.07.)

a. Closing a file with NO REWIND is not implemented.

b. FSPACE spaces on block rather than records.

c. Chained SIOs on mag tape do not perform correctly causing transfer larger than 4095 words to fail when record length is undefined.

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

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

f. Input arguments to BINARY of [±] 65536, 65537, 65538, and 65539 do ***not*** return overflow.

g. The FOPEN's FORMSG parameter is a byte array: if this array begins on odd byte boundary, then preceding byte is also printed.

h. Lower case :EOD is not recognized on a data accepting device. Only upper case letters are recognized for that command.

i. The command :DEALLOCATE of a non-existing program file comes back with ERR 217. It should read ERR 217, 52 where 52 is the corresponding FCHECK number. (The file referenced is not in the system file domain.)

j. $STDLIST Allocation Problem

When the "max # of open spoofles" (SYSDUMP /INITIAL) are not sufficient to adequately handle all spooling requirements (see below), spooled JOBs to line printers (including :STREAM) may cause "endless" numbers of null list files to be generated in certain situations.

This bug manifests itself as multiple $STDLIST files for a single JOB, each producing simply a header and trailer with no real JOB output: null files. If the list line printer is spooled, this results in many null spoofles (each four sectors); in the case of non-spooled line printers, the headers/trailers are printed immediately—if such non-spooled line printers are offline, the system will crash due to IOQ overflow in this situation. If an open spoofle is closed during this "resource loopiñg" by some running user, the provoking JOB may finally get launched normally. In this case, the last $STDLIST file for JOB will be the true job listing.

This bug can be overcome by increasing the open spoofle limit. The recommended value is around 20. A better estimate may be obtained by considering that the ***initial*** allocation (FOPEN) of a spooled device for ***user*** access creates a newly opened spoofle. On the corresponding FCLOSE, the spoofle becomes unopened.

For example:

A SESSION's single access to a spooled line printer results in one open spoofle; a spooled JOB to a spooled list device requires at least two—however, each further access to $STDIN or $STDLIST within the JOB "reuses" the already allocated ones; each of the JOB's FOPEN's of "LP" though requires another one. Expected concurrency is important because the limit applies to those spoofles open at the same time. An indication that the limit is being pressed is failures on allocations to spooled devices, assuming the spool discs are healthy and have sufficient space.

VI A. Documentation Changes (MPE CO.06)

1. MPE 3000 Reference Manual (32000-90002)

pg. 8-60

Flags

A logical value which may be used to specify options as follows:

(14:1) = 1	Suppress ASCII Dump
(15:1) = 1	Suppress Traceback of Stack Markers

If omitted, the default value for flags is 0.

pg. 8-65

Flags

(15:1)	A DL to Qi Dump (see Figure 8-1)
(14:1)	A Qi to S Dump
(13:1)	A Q-63 to S Dump (ignored if flags (14:1) is on)
(12:1)	(Ignored)
(11:1)	Suppress ASCII portion of dump

pg. 8-32

(Line before last should read:)

"refer to NOTE ..... page 8-30);"

pg. 8-18

Following the specification of the intrinsic CHRONOS, insert specifications of intrinsics

CLOCK
and
CALENDAR

as they appear in this article under Enhancements to MPE CO.06.

2. MPE 3000 Operator's Guide (32000-90004)

   pg. 3-25

   The area of bad track recovery is not clearly explained in the manual. For more precision, refer to Operational Notes in this article.

VI. B. Documentation Changes (MPE CO.07)

1. MPE C.00.05
   (May 7, 1975) – Note Erratum

   pg. 16

   The actual command to run SDUP should read:

   ```
   :RUN SDUP.HPOFFLN.SUPPORT;
   NOPRIV
   ```

   Failure to do so would result in an abortion of the program.

2. MPE/3000 Operating System Manual (Part #32000-90002)

   pg. 10-26

   New FCHECK error numbers (106 and 107) were introduced with MPE C.00.04. See MPE Note dated April 14, 1975.

3. MPE/3000 Operating System, System Manager/System Supervisor Manual (Part #32000-90006)

   pg. E-1

   In the Configuration Value Table, the maximum for the directory is given to be 512 sectors. In fact, the maximum (regardless of disc space) is 6000 sectors.

   pg. 5-28

   Step No. (h) replaced with:

   ```
   FOLLOWING FILES PURGED
   BECAUSE OF:
   INSUFFICIENT DISC SPACE (1)
   TAPE PARITY ERROR (2)
   LIST?
   ```

   To list those files that were not reloaded because:

   (1) not enough disc space was available; or (2) a Tape Parity Error was encountered while reading the file from tape, enter YES.

   To suppress the listing, enter NO.

4. MPE/3000 Operator's Guide (Part #32000-90004)

   pg. 8-13/8-14

   All "ININ (12)" halts have been removed and replaced with system failures.

   pg. 8-3

   Add System Failure Numbers:

130	ININ PARITYERR	Memory Address Parity Error
131	ININ PARITYERR	System Parity Error
132	ININ PARITYERR	Irrecoverable Memory Parity Error

VII. Operational Notes

A. Recovering from a bad track on the disc.

This discussion attmepts to clarify some aspects of recovery when the system has detected a bad track while running (generally through a SYSTEM FAILURE #52).

Glossary
In the portion of the INITIAL dialogue concerning the suspect disctrack, the meaning of the terms RECOVER/DELETE/REASSIGN is somewhat obscure. When a bad track is discovered while the system is running, the corresponding track address and logical device number is stored in the Defective Track Table and will be, on subsequent warmstart or coolstart, recognized as a suspect track. At that point, the user has different options.

RECOVER
Means recover space from the Defective Track Table, delete the entry, and forget that this track had ever been suspect.

DELETE
Means delete that suspect track from the usable portion of the disc. If the track logically belongs to a sensitive area of the system (e.g., System Directory), it is only possible to delete it on a RELOAD, to give the system a chance to reconstruct the proper data around the bad track.

REASSIGN
(For Moving Head Disc only.) An alternate track on the disc is taken and linked to the bad track. Every time the bad track is accessed, such an indication is given to the driver and the alternate track is accessed instead.

It follows that if a bad track is detected in the Directory on a cool or warmstart, then a mention of the suspect track should be left in the Defective Track Table.

Example:
Suppose that a suspect track is detected on a warmstart and that it happens to reside in the Directory. The following message would appear:

```
**WARNING**DIRECTORY
RECOVER?
```

To keep the track address in Defective Track Table, answer ***NO***.

System resumes cold load. When it is up, it has to be brought down and RELOADed. The message then appears:

```
**WARNING**DIRECTORY
DELETE OR RECOVER?
```

This time the answer is DELETE in order to remove that track from the usable portion of the disc. The Directory is then rebuilt.

B. Tape Low Condition for FWRITEs to Paper Tape Punches

The FWRITE intrinsic, with cooperation from the Paper Tape Punch driver IOPTPNO, has been modified so that it will now cause special handling of the EOT (Tape Low) condition when the device is a Paper Tape Punch.

Previously, CCL was returned by FWRITE when EOT was sensed by any device. FWRITE will now make a distinction between EOTs from Paper Tape Punches and other devices in the following way.

EOT from a Paper Tape Punch (Tape Low) will cause FWRITE to:

(1) Pur.ch a trailer on the paper tape.
(2) Emit the message:

?IO/<time>/# $\overset{J}{S}$ <number>/<PIN>/TAPE LOW LDEV#<ldev>. IS TAPE MOUNTED? (Y/N)

and suspend, waiting for the operator to mount a fresh reel of paper tape and/or respond to the message.

(3) If the response is "Y", a header is punched and a CCE is returned making the EOT condition transparent to the user.

(4) If the response is "N", no header is punched and a CCL, as before, is returned.

C. FORTRAN and HP 2984A Users

There is a problem associated with opening FORTRAN files to the HP 2984A Card Reader/Punch Subsystem. The result of the problem can cause data read by a FORTRAN program to be incorrect.

When a file is opened by FORTRAN to the card reader/punch, it is opened with variable record format. This happens because the card reader/punch has a device type which specifies serial I/O. Therefore, the initial byte of a file opened in this manner is used by the file system to indicate the size of the record to be read. This is what causes the FORTRAN data to be read incorrectly.

To get around the problem, users of FORTRAN and the 2984A Card Reader/Punch should specify either NOBUF or undefined record length in their file commands. In either case, data will be read with undefined record format, and thus alleviate the problem of the initial byte being interpreted as record length.

Examples:

```
#1  :FILE FTN05=$STDIN;NOBUF
    :FILE FTN50=$STDIN
      .
      .
      .
    READ(5),10)IJ
    READ(50,13)L

#2  :FILE FTN05=$STDIN;NOBUF
    :FILE FTN50=$STDIN;NOBUF
      .
      .
      .
    READ(5,10)IJ
    READ(50,13)L
```

```
DATA

 1   2   5   4   3
10  23  23  55  55
```

Both examples will read data from the first card correctly. However, the first example will not read the second data card properly. It is here that the first byte is taken as the record length. The second example reads both data cards correctly :FILE FTN50=$STDIN;REC=,,U could also be used with the second example.

HP 32100A SPL/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32100A.05.01 SPL/3000. The MIT date coded 1525 did not contain any updates to SPL/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View, California.

Part No. 32100-1X002 SPL MAINT.

There are no changes to be made to the documentation as a result of this release.

Problems Corrected in SPL/3000.05.01

1. Initialization of a byte pointer to a direct array address was incorrect.
2. A call to PRINT'FILE'INFO is made when a file cannot be opened.
3. Some internal tables were incorrect when the symbol table exceeded 16383 words.
4. An external variable used as the recipient in a deposit field caused incorrect code to be generated.
5. An error message is emitted if a code module exceeds 16383 words.
6. A compare range and branch instruction if emitted just as an internal table was full resulted in a bounds violation.
7. A warning message is emitted if SUBPROGRAM and USLINIT are both specified.
8. A MOVE statement in the range of a FOR statement containing formal parameters which are S-relative resulted in incorrect addresses being used.
9. Assemble statements in subprogram mode sometimes resulted in the location counter alongside the source output to be incorrect.
10. Code to load the X-register for an array indexed by a variable was missing under certain conditions.
11. CALENDAR and CLOCK are called in place of CHRONOS.

Note:

Since 10 above could have adversely affected our products, all products compiled with SPL 05.00 were recompiled and verified. The only one affected was the COMPILER LIBRARY, which was detected and corrected on the MIT date coded 1525.

HP32101A BASIC/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32101A.03.01 BASIC/3000. There were no updates to BASIC/3000 on MIT date coded 1525.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32101-1X002 BASIC MAINT.

No changes to the documentation are required as a result of this release.

Problems Corrected in BASIC/3000.03.01

1. An end-of-file on the INVOKE file is no longer detected when reverting from an INVOKE to a program with no non-COM variables.
2. An INVOKE file error during reversion left the interpreter in an improper state.
3. Bad byte address was generated when evaluating a string variable in large-stack situations; resulted in a bounds violation.
4. Message "MISSING 'THEN' " has been changed to "ILLEGAL EXPRESSION OR MISSING 'THEN' ".
5. Message "EMPTY PROGRAM NAME" has been changed to "PROGRAM NAME IS UNDEFINED".
6. MPE-T-dependent code in the interpreter has been removed. (BASIC should now be :PREP'ed with IA, BA, and PH capabilities.)
7. INVOKE no longer modifies DB+0; this occasionally caused a stack underflow.

Problems That Still Exist in BASIC/3000.03.01

1. $NULL is always treated as a binary file. This causes ADVANCE and LINPUT to terminate with an error.
2. PRINT #F;END is not handled correctly for 1-record files.
3. Typing control-Y and using ABORT, CALLS, or FILES in break mode sometimes causes a stack underflow abort. This problem can be avoided by using the following procedure:

   a. Type control-Y; the break message indicates the current statement number.

b. Set a break point at the next statement.

c. Type GO.

d. BASIC will break at the break point; it is now safe to ABORT or use CALLS or FILES.

4. PRINT #F; "" and PRINT "" transmit nothing to the file instead of an "empty" record.

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

HP 32103A BASIC COMPILER/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32103A.00.02 BASIC COMPILER/3000. There were no updates to BASIC COMPILER/3000 on the MIT date coded 1525.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32103-1X002 BASICOMP MAINT.

There are no changes to the documentation as a result of this release.

Problems Corrected in BASIC COMPILER/3000.00.02

1. Indirect simple variables (e.g., LONG's) were incorrectly allocated when more than one required Q-negative address space. This commonly resulted in a CHAIN/INVOKE error #46.

2. When allocating string arrays in COMmon, the maximum-size word was incorrectly stored. This had no adverse effect on programs that remained entirely in BASIC.

3. A LET statement with only one COMPLEX destination variable caused spurious stack cut-back. This yielded anomalous results (e.g., early termination of FOR-loops).

4. COMPLEX CSH and SNH yielded incorrect results.

5. Unary plus ("+") caused its operand to be ignored in the expression evaluation, yielding anomalous results.

6. In programs compiled with the INIT option and requiring Q-negative address space (i.e., global variables requiring more than 124 words), the stack marker was incorrectly relocated. This resulted in a spurious CHAIN/INVOKE error #46.

7. When the recsize parameter of the CREATE statement was not a constant or INTEGER variable, the parameter was incorrectly evaluated. This resulted in a record size error #28.

8. A numeric ENTER statement (i.e., numeric destination variable) clobbered several locations above the destination variable.

9. When there was insufficient stack for buffers for files declared in a FILES statement (error #37), a spurious "FILE #0 UNDEFINED" message was displayed.

10. Due to a typo, literal strings on the PURGE statement were not handled optimally. This had no adverse effect.

11. Very long IMAGE statements and literal format strings (~ 120 bytes) caused bounds violation abort at run-time.

12. An INPUT statement whose I/O list was terminated by a FOR-loop clobbered stack and caused spurious stack cut-back. This yielded anomalous results (e.g., early termination of enclosing FOR-loop).

Problems That Still Exist in BASIC COMPILER/3000.00.02

1. Unary minus ("–") preceding a constant is not handled according to the precedence rules. This will yield different results than the interpreter in two cases:

   a. "–3 MOD x" is evaluated as "(–3) MOD x" instead of "–(3 MOD x)"
   b. "–1**x" is evaluated as "(–1)**x" instead of "–(1**x)"

   As a temporary work-around, fully parenthesize expression or substitute variables for the constants.

2. 0**(--2L0) yields a single-precision "infinity".

3. Lower-case characters are not recognized as format specifications in format strings.

HP 32213 COBOL-A
HP 32213 COBOL-B

This article, along with the release of the MIT tape date coded 1525, comprises the official release of update level 02.00 (COBOL-A) and 01.00 (COBOL-B) for the subsystem HP 32213 COBOL/3000.

Problems Corrected in COBOL-A.02.00

1.0 DYNAMIC subprograms containing a VALUE clause with a non-numeric literal exceeding 116 characters cause error #201 at compile time.

1.1 SET-TO statement where the object is a table-element appearing in a sub-program generates incorrect code. (i.e., subscript or index is ignored.)

1.2 Complex IF statements containing ORs followed by AND generate incorrect code. (i.e., the AND is ignored.)

1.3 Programs with USING procedures will QUIT after executing procedure.

Problems Corrected in COBOL-B.02.00

2.0 MOVE ZERO TO X, where X is COMP-3 item leaves redundant word on run-time stack.

2.1 DYNAMIC subprograms containing a VALUE clause with a non-numeric literal exceeding 116 characters cause error #201 at compile time.

2.2 Incorrect code is generated for an ADD-TO statement in which both operands are of COMP-3 type and the result identifier is a table-element.

2.3 PERFORM-UNTIL statements may generate incorrect code resulting in looping at object time.

2.4 SORT statements with input procedure or non-disc input file are limited to 10,000 records. This is changed as follows:

> if SORT-file size < 100 records, then 10,000 records is assumed else the SORT-file size is taken as the maximum number of records to be sorted.

2.5 Arithmetic statements with ON SIZE ERROR and result identifier of COMP-3 type generate error #211.

2.6 Programs with USING procedures will QUIT after executing procedure.

Problems Corrected in COBOL-A.02.01.

1.1 Unsigned arithmetic data items now receive absolute values of results from MOVE or arithmetic statements in all cases.

1.2 Parameter byte-addresses > 32K in CALL statements are now correct.

1.3 An error diagnostic is now generated for a MOVE to a paragraph-name or section-name.

1.4 A comparison of a group item with HIGH-VALUES failed if the group-item had a COMP-3 declaration attached to it.

1.5 Legal pictures of all floating – , +, or $ were diagnosed as errors.

Problems Corrected in COBOL-B.02.01.

2.1 Unsigned arithmetic data items now receive absolute values of results from MOVE or arithmetic statements in all cases.

2.2 Parameter byte-addresses > 32K in CALL statements are now correct.

2.3 An error diagnostic is now generated for a MOVE to a paragraph-name or section-name.

2.4 COMPUTE statement with operands of COMP type now executes correctly.

2.5 DIVIDE statement with REMAINDER which has more digits in the resultant operand than in the quotient now executes correctly.

2.6 A comparison of a group item with HIGH-VALUES failed if the group-item had a COMP-3 declaration attached to it.

2.7 A COMPUTE statement with a division of one-word COMP items failed.

2.8 Numeric MOVE statements with a table-element as the sending operand and more than one receiving operands failed for the second and following operands.

2.9 Legal pictures of all floating +, –, or $ were diagnosed as errors.

HP 32213 COBOL-A
HP 32213 COBOL-B

This article, along with the MIT date coded 1531, comprises the official release of HP 32213.02.01 (COBOL-A) and 01.01 (COBOL-B) COBOL/3000.

HP 32104A RPG/3000

This article, along with the release of the MIT date coded 1525, comprises the official release of fix level .06 for the subsystem HP 32104A.01.06 RPG/3000.

Incorporated in this fix level are corrections of the following problems having occurred in the previous release of HP 32104A.01.03.

1.0 SEGMENTER goes into loop while preparing RPG generated USL.

1.1 Generate PCAL instead of LLBL when RPG tries to get user procedures PLABEL.

1.2 Does not inform user when error occurs in segmented write run-time procedure.

1.3 Terminate when user defines a DEBUG listing file record size other than 80 characters long.

1.4 Does not inform user when IMAGE/3000 is not on system SL when user tries to use RPG/IMAGE interface.

1.5 OF and 1P indicators cause too many heading lines printed.

1.6 Pass wrong parameters to user's SPECIAL file procedure.

1.7 MOVEL generates the wrong result.

1.8 Updating an IMAGE file does not work.

Note: There was ***no*** release for fix level .04 or .05, therefore, the previous update and fix level was 01.03 and the next update and fix level is 01.06.

HP 32104A RPG/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32104A.01.07 RPG/3000.

Incorporated in this fix level are corrections of the following problems having occurred in the previous release of HP 32104A.01.06.

1.0 TESTN may destroy data.

1.1 Tally Line Printer skips only 15 lines (so skips > 15 lines did not work).

1.2 End of file on one file may cause end of job.

1.3 Arrays with even number of digits cannot be outputted with literal subscript.

1.4 For blank on all space and skip output entries, compiler assumes space one before—will now assume space one after.

1.5 Record Identification of column past 512 does not work.

1.6 No field descriptions on output gives terminal compiler error.

1.7 Update adds are now done to end of file.

1.8 Chaining to output file—file was initialized.

1.9 If more than one Image data base in mode 1, there may be a locking error.

1.10 MOVEL problem.

HP 32215A IMAGE/3000

This release of IMAGE/3000, HP 32215A.02.01, along with the MIT date coded 1525, is corrected and enhanced to reflect the following:

- DBOPEN will now accept in the BASE parameter a fully qualified data base name, that is, one which includes an account name (i.e., BASE.GROUP. ACCOUNT). In this way a user program can, if desired, access data bases across accounts; that is, data bases in accounts other than the user's log-on account. All such access is, of course, subject to all MPE file system account, group, and file security provisions.

- The special list construct "@;" may now be used by a program which has ***read*** access to all data items in the referenced data set. This list was formerly restricted to programs whose access level was greater than or equal to the ***write*** level of the data set. In its extended use, the "@;" list is treated as "shorthand" for "all data items in the entry", and is processed exactly as if all items in the data set were explicitly given, in order, in the list parameter.

  This change may be of particular interest to users of RPG, which always uses "@;" in its calls to IMAGE intrinsics requiring a list parameter.

The MIT date coded 1531 contains no updates to IMAGE/3000.

HP 32216A QUERY/3000

This article, along with the release of the MIT date coded 1525, comprises the official release of subsystem HP 32216A.01.00 QUERY/3000.

Incorporated in this update level is the correction of the following problems having occurred in the previous release of HP 32216A.00.01, this update also reflects enhancements as stated below:

The message "INTEGER VALUE ERROR" has been changed to "NUMERIC VALUE ERROR".

The following messages have been added:

1. ILLEGAL REPORT CONTROL STATEMENT

   The report control statement OUT=LP has bad format.

2. ILLEGAL ASCENDING/DESCENDING CODE

   The ASC or DES parameter is missing or misspelled.

3. END OF XEQ FILE

   Output when an end-of-file occurs on an XEQ file.

4. UNABLE TO USE FILE

   The requested new PROC-FILE name is not a new file or an empty existing PROC-FILE.

The following information will discuss the changes to each of the command categories:

I. General
II. Environment
III. Locating
IV. Reporting
V. Updating
VI. Procedure
VII. Utility

I. General

Change the QUERY prompt to mean:

>	prompt for command input
>>	prompt for additional command parameters or continuation line (previous line terminated by a &)

This would eliminate the printing of the character string "NEXT?".

A session user of QUERY will no longer be prompted for the operating environment information. The user must issue the required command(s) to establish his/her environment.

Correct the specification of the following data item types handled by QUERY:

–ASCII character string	(Un)	for	(1<=n<=255)
–ASCII character string	(Xn)	for	(1<=n<=255)
–Zoned decimal numbers	(Zn)	for	(1<=n<=255)
–Packed decimal numbers	(Pn)	for	(2<=n<=254) (n must be even)

II. Environment

DATA-BASE

This command is used to change the data base being accessed. DATA-BASE will cause prompts for LEVELWORD = and MODE = to be answered.

A user may also change his level or mode of access by issuing the new LEVELWORD= or MODE= commands. If either LEVELWORD= or MODE= is entered, then data base name must have been entered previously.

DATA-SETS

No change.

PROC-FILE

Add an optional parameter after filename specifying file size.

PROC-FILE = filename [,N]

where 5 <= N <= 400
default N = 126

If the filename does not exist, then QUERY will issue a

FILE DOES NOT EXIST, BEING CREATED

message and open and save a disc file with the specified filename using a file code of 1070 and of size N.

The user will no longer have to BUILD a procedure file.

A file code of 1070 has been assigned to QUERY PROC-FILEs.

If a PROC-FILE filename specified currently exists in the system file domain, it will be opened. If the filename does not have a file code of 1070, the following messages are output:

```
BAD FORMAT ON PROCEDURE FILE
DO YOU WISH TO CONVERT PROC-FILE
>> [YES/NO]
```

If answered YES, then the user will be prompted for a new filename with:

NEW PROC-FILE NAME =

The new filename will be opened and the old procedure file will be converted into the new procedure file (see *Procedure* Section).

The specified new filename, if it exists, must be an empty procedure file.

The user may PURGE the old PROC-FILE, if desired, at this time.

OUTPUT

No change.

DEFINE

With the addition of the LEVELWORD and MODE commands, these items will be displayed when DEFINE is entered.

A session user of QUERY will no longer be prompted for the operating environment information. The user must issue the required command(s) to establish his/her environment.

III. Locating

New relational operators:

EQ,NE,LT,GT,LE,GE

The message

SERIAL READ MUST BE PERFORMED, CONTINUE (YES OR NO)?

is replaced with

USING SERIAL READ

To abort the FIND Command, use CTRL Y.

New format for invoking FIND commands from a PROC-FILE IS

FIND procedure name [,character]

where cahracter is any ASCII character.

If character is included in the command, then the FIND procedure is listed. If character does not appear, then the procedure is not listed.

IV. Reporting

REPORT ALL will display each sub-item which has a sub-item count greater than one (1).

HEADER STATEMENT

The date type in a Header statement need not be a master data item.

SORT STATEMENT

New optional parameter [,ASC/DES].

The DES option, if specified, indicates that the data item values are to be ordered in descending order. If DES is not specified, or ASC is, the data item values are ordered in ascending order.

DETAIL STATEMENT

No change

GROUP STATEMENT

No change

TOTAL STATEMENT

The arithmetic options of ADD and AVERAGE will not be allowed on ASCII (U or X) data types.

EDIT STATEMENT

No change.

OUTPUT CONTROL

Report output control parameters may be specified at any place in a REPORT command, but, if specified, must precede the keyword ALL.

LINES = integer

Integer specifies the number of lines per page for a report. The default is 60. If changed by the user, it will be reset at the end of each report command to the default value.

The LINES = controls whether the page size is of infinite length (LINES = 0) or a fixed length.

Integer must be 0 or between 10 and 32767

NOPAGE

Specifies that page advancing is not to be done on each report page and before report output is begun. The default is to page advance.

OUT=LP

Will switch output to the QSLIST device for the current report only.

PAUSE

If specified, report output will pause after each page has been completed; hit carriage return to continue. PAUSE is ignored if job is not interactive or output is being sent to the QSLIST device.

V. Updating

Each sub-command of UPDATE (ADD, DELETE, and REPLACE) will now be recognized as a valid command.

Invoking any UPDATE procedures must be done using

UPDATE procedure name

and will have an additional optional parameter [,character] following the procedure name. If character is included in the command, then the UPDATE procedure is listed. If character does not appear, then the procedure is not listed.

VI. Procedure

Procedure names may be eight (8) characters in length. PROC-FILEs containing six (6) character procedure names will be converted by QUERY to eight characters when the old PROC-FILE is specified.

ALTER

No change.

CREATE

Add PAUSE and NOPAGE to the list of illegal procedure names.

Filename in a CREATE command must not be FIND, REPORT, or UPDATE or any substring of these command words.

The last eight (8) characters of each filename record are not processed by QUERY.

The CREATE SPACE command output now is

RECORDS =

instead of SECTORS.

If a procedure currently being created overflows, the available space left in the PROC-FILE input is terminated with an error message. The incomplete procedure is stored and may be listed by using the DISPLAY command.

DESTROY

No change.

DISPLAY

If the DISPLAY procedure name, filename form of the command is used and filename does not exist in the log-on group, the file will be created by QUERY and saved using filename as the formal designator with

FILE DOES NOT EXIST, BEING CREATED

output to the user. The file size will be 200 records.

A DISPLAY LIST output to the QSLIST device will include a header line giving the procedure file name and a date stamp.

RENAME

The RENAME command will change the name of a given procedure in the current PROC-FILE.

FORMAT:	RENAME current name,newname
EXAMPLE:	RENAME OLD,NEW

VII. Utility

EXIT

No change

FORM

Add the following optional parameters to the FORM command format:

data item name
SETS
ITEMS
PATHS

Information produced for each of the above elements is:

data item name – Item name, type, sub-item count and length, read and write levels, and name of each data set in which the specificd item occurs.

SETS – Set name, type, capacity, entry count, entry length, and blocking factor of each data set in the data base.

ITEMS – Item name, type, sub-item count and length, read and write levels of each data item in the data base.

PATHS - Relationship between data sets in the data base.

If a data set and data item have the same name, the data set name will be processed.

If a data set or data item name is identical to the parameter keywords of SETS, ITEMS, or PATHS, it will not be processed if specified as a FORM command parameter.

HELP

No change.

XEQ

Will cause QUERY to read a file containing commands and command parameters.

FORMAT: XEQ filename

where filename is an ASCII file containing commands and parameters.

When the XEQ command is entered, the specified file is read and the commands executed until an end-of-file is reached or another XEQ command. Any command input is also read from filename. When an end-of-file on filename is reached, control returns to the original command file ($STDIN).

If an error occurs attempting to open a data base, for a session user, the XEQ file is closed and further input is read from $STDIN.

An XEQ command within the XEQ file will close the first file and open the new one.

Only the first 72 characters of each record are read.

A session user reading an XEQ file is considered to be not interactive.

VERSION

FORMAT: VERSION

This command will output the latest version, update, and fix level information on QUERY and all IMAGE program files and procedures. This information is useful when submitting program information to HP support personnel.

There is no correspondence between QUERY and IMAGE version, update, or fix levels. All IMAGE programs and procedures may not have the same fix level but should have the same update level.

Note: There are no changes in the System Library.

DOCUMENTATION CHANGES (QUERY/3000, MPE C0.06)

These are guidelines to be used until official update of manual occurs.

Page 1-2

Data Types

– Zoned decimal numbers (Zn) $(1 \leq n \leq 255)$

– Packed decimal number (Pn) $(2 \leq n \leq 254)$ (n must be even)

– ASCII character strings (Un) $(1 \leq n \leq 255)$

– General ASCII character string (Xn) $(1 \leq n \leq 255)$

Page 1-5

Figure 1-3 – Change LEVEL to LEVELWORD

Page 1-6

Table 1-1 – Add

•ADD	Adds data entries to the data base
•DELETE	Removes data entries from the data base
•LEVELWORD	Changes access level word
•MODE	Changes access mode
•RENAME	Changes procedure name
•REPLACE	Modifies data items in data entries
•VERSION	Displays version update and fix information
•XEQ	Reads command file

Page 1-8

Table 1-3

>	Prompt for command input
>>	Prompt for additional command parameters or continuation line (previous line terminated by a &).

Page 2-1

Part 'b'

Delete as presently existing.
Replace with "The user no longer has to BUILD a procedure file."

Part 'c'

QUERY no longer prompts you for the operating environment after being invoked. It is your responsibility to issue the required command(s) to establish your environment.

Page 2-2

Paragraph 5

Change: "Your response may be the name..."
To: "Your response may be the name of a new file or an already existing procedure file containing QUERY procedures stored there from previous QUERY executions."

Page 2-3

Part 'd'

Delete: "the character string"
"NEXT?"
"or some other"

Page 2-4

Paragraph 2

Delete: "Unlike session mode,"

Page 3-2

Table 3-3 – Add

Environment

LEVELWORD =	Specifies access level word
MODE =	Specifies access mode

Updating

ADD	Adds data entries to the data base.
DELETE	Removes data entries from the data base
REPLACE	Modifies data items in data entries

Procedure	
RENAME	Changes procedure name
Utility	
VERSION	Display version, update, and fix information.
XEQ	Reads command file

Page 3-3

Data Base Name

Delete: "stored in the log-on account. (The log-on.....to MPE)."

Page 3-3

Paragraph 3-8

Change LEVEL to LEVELWORD

Page 3-4

Paragraph 3-9, Figure 3-1

After "INVALID MODE", delete:

"NEXT?"
">DATA BASE=PROJ"
"LEVEL => MANAGER"

Page 3-4

After Paragraph 3-9, add

A user may also change his level or mode of access by issuing the LEVELWORD= or MODE= commands. If either command is entered, QUERY first closes the current data base before attempting to open the data base with either the new level word or access mode.

Page 3-8

Paragraph 3-17

Add: An optional parameter after filename specifying file size [,N].

where $5 \leqslant N \leqslant 400$
default N=126

Delete: "In session mode, ... 5 and 125, inclusive,"

Add:

If the filename does not exist, then QUERY will issue a "FILE DOES NOT EXIST, BEING CREATED" message and open and save a disc file with the specified filename using a file code of 1070 and of size N.

The user will no longer have to BUILD a procedure file.

All QUERY PROC-FILEs have been assigned a file code of 1070.

If a PROC-FILE filename specified currently exists in the system file domain, it will be opened. If the filename does not have a file code of 1070, the following are output:

```
BAD FORMAT ON PROCEDURE FILE
DO YOU WISH TO CONVERT PROC-FILE?
>>
```

If answered YES, then the user will be prompted for a new filename with:

```
NEW PROC-FILE NAME =
```

The new filename will be opened and the old procedure file will be converted into the new procedure file.

If the specified new filename exists, it must be an empty file or QUERY outputs an error message.

Page 3-9

Third paragraph

Change "sector" to "record".

Page 3-13

Delete: "If you do not change . . . . or access mode."
"The prompt and message . . . . for further details."

Page 3-14

Paragraph 3-29

Delete: "This causes QUERY.....access mode."

Page 3-16

Value

Delete:

"As shown above . . . . . in quote marks (")."

Add:

Value need not be enclosed in quote marks (") unless the value contains special characters or blanks.

Page 3-17

Table 3-2

Add:

EQ	is equal to
NE	is not equal to
LT	is less than
GT	is greater than
LE	is less than or equal to
GE	is greater than or equal to

Replace "SERIAL SEARCH . . . . . . (YES or NO)" with "USING SERIAL READ".

Page 3-22

Paragraph 3-37

Change "FIND procedure name" to "FIND procedure name[,character]", where character is any printing character.

Change "SERIAL......(YES or NO)" to "USING SERIAL READ".

Delete: "To perform . . . . . . another command."

Add: If character is included in the command, then the FIND procedure is listed. If character does not appear, then the procedure is not listed.

Page 3-29

"data type"

The data item name need not be from a master data set.

Page 3-31

New optional parameter

[,ASC/DES]

The DES option, if specified, indicates that the data item values are to be ordered in descending order.

Page 3-34

DETAIL STATEMENT

Data Number

Change "integer from 1 to 5" to "integer from 1 to 9".

Page 3-39

"data type"

Add: The arithmetic options of ADD and AVERAGE will not be allowed on ASCII (U or X) data types.

Page 3-47

Add after 3-53

Report Output Control Statements

LINES = integer

Integer specifies the number of lines per page for a report. The default is 60.

The LINES = controls whether the page size is of infinite length (LINES = 0) or a fixed length.

Integer must be 0 or between 10 and 32767.

NOPAGE

Specifies that page advancing is not to be done on each report page and before report output is begun. The default is to page advance.

OUT=LP

Will switch output to the QSLIST device for the current report only.

PAUSE

If specified, report output will pause after each page has been completed. Hit carriage return to continue.

Update Commands

Each sub-command of UPDATE (ADD, DELETE, and REPLACE) will now be recognized as a valid command.

Invoking *any* UPDATE Procedure must be done using:

UPDATE procedure name[,character]

If character is included in the command, then the UPDATE procedure is listed. If character does not appear, then the procedure is not listed.

Page 3-57

'Procedure Commands'

Delete: "The file may be . . . . . . of your choice."

Add: RENAME changes name of a given procedure.

Change: "unused sectors" to "unused records".

Page 3-58

"procedure name"

Change "six characters" to "eight characters".

Change "LIST, SPACE" to "LIST,PAUSE,NOPAGE, SPACE".

Paragraph 3-74

Change all "sector" to "record"; "sectors" to "records".

Page 3-59

Add:

Filename in a CREATE command must not be FIND, REPORT, or UPDATE or any substring of these command words.

The last eight (8) characters of each filename record are not processed.

Page 3-60

Change a ">" to a ">>".

Page 3-61

Paragraph 3-80

Add:

If the filename does not exist in the log-on group, the file will be created by QUERY and saved using filename as the formal designator with

FILE DOES NOT EXIST, BEING CREATED

output. The file size will be 200 records.

Page 3-67

Add after Figure 3-31

3-95A. RENAME

3-95B. FUNCTION. To rename a procedure in the current PROC-FILE.

3-95C. FORMAT. The form of the command is:

$\left\{\begin{array}{l}\text{RENAME}\\ \text{REN}\end{array}\right\}$ old procedure name, new procedure name

Page 3-67

Paragraph 3-99

$$\left\{\begin{array}{l}\text{FORM}\\ \text{FO}\end{array}\right\} \qquad \left[\begin{array}{l}\text{data set name}\\ \text{data item name}\\ \text{SETS}\\ \text{ITEMS}\\ \text{PATHS}\end{array}\right]$$

Information produced for each of the new optional parameters is:

data item name –
item name, type, sub-item count and length, read and write levels, and name of each data set in which the specified item occurs.

SETS –
set name, type, capacity, entry count, entry length, and blocking factor of each data set in the data base.

ITEMS –
item name, type, sub-item count and length, read and write levels of each data item in the data base.

PATHS –
relationship between data sets in the data base.

Page 3-73

Add after paragraph 3-111

VERSION
FORMAT: VERSION

This command will output the current version, update, and fix level information on QUERY and all IMAGE program files and procedures.

There is no correspondence between QUERY and IMAGE version, update, or fix level.

XEQ
FORMAT: XEQ filename

Where filename is an ASCII file containing commands and parameters.

When the XEQ command is entered, the specified file is read and the commands executed until an end-of-file is reached or another XEQ command. Any command input is also read from file-name. When an end-of-file on filename is reached, control returns to the original command file ($STDIN). Only the first 72 characters of each record are read.

Page 5-2

Delete: "NEXT?"

Page 5-4

Change "INTEGER VALUE ERROR" to "NUMERIC VALUE ERROR".

Page 5-6

Change "none of the overflowed procedure is stored" to "As much of the overflowed procedure as possible is stored. List the procedure using the DISPLAY command to determine how much was saved."

Change "SECTORS" to "RECORDS" and "unused sectors" to "unused records".

Page 5-7

Change "LEVEL ERROR" to "LEVELWORD ERROR".

Page 5-12

Change "SERIAL READ....." to "USING SERIAL READ".

Page 5-13

PROC-FILE

Add: "UNABLE TO USE FILE"
The requested new PROC-FILE name is not a new file nor an empty existing PROC-FILE.

Page 5-16

Add: ILLEGAL ASCENDING/DESCENDING CODE
The ASC or DES SORT parameter is missing or incorrectly spelled.
ILLEGAL REPORT CONTROL STATEMENT
The report control statement OUT=LP has bad format.

Page 5-21

Change "INTEGER VALUE ERROR......" to "NUMERIC VALUE.....".

Page 5-22

Add: END OF XEQ FILE
Output when an end-of-file occurs on an XEQ file.

HP 32216A QUERY/3000

This article, along with the release of the MIT date coded 1531, comprises the official release of HP 32216A.01.01 QUERY/3000.

Incorporated in this fix level is the correction of the following problems having occurred in the previous release of HP 32216A.01.00; this update also reflects enhancements as stated below.

1.0 Problems Corrected:

1.0 Handling of control characters in the access LEVELWORD.

1.2 A FIND with "Z" or "P" data types did not always locate the requested entry.

1.3 An UPDATE ADD to a data set with more than 99 data items resulted in "ILLEGAL ACCESS".

1.4 The REPORT ADD option with real data types (R2 or R3) produced erroneous results.

2.0 Changes Made:

2.1 Correct the initialization of the PROC-FILE. This eliminates the possible "FILE READ ERROR 0" in ALTER

2.2 Leading zeros in "Z" or "P" data types are now suppressed in a REPORT body.

2.3 Change the following error messages:

AUTOMATIC MASTER IS FULL
CHAIN HEAD IS FULL
NO MANUAL ENTRY FOR DETAIL

to:

FULL MASTER FOR ***item/set***
FULL CHAIN FOR ***item/set***
MISSING CHAIN HEAD FOR ***item/set***

where:

item is the detail data set search item name, and
set is the related master data set name

3.0 Documentation Changes (Query/3000, MPE CO.07)

3.1 Page 3-51, Figure 3-22

Change:
"NO MANUAL ENTRY FOR DETAIL"
To:
"MISSING CHAIN HEAD FOR PROJECT/ PROJECT-M"

3.2 Page 5-19, after "AUTOMATIC MASTER" error

Add:

FULL MASTER FOR ***item/set***

The user has attempted to add a detail data entry with a search item value (that does not match any existing search item value) in the corresponding automatic master data set, and a new master entry cannot be created because the automatic master data set is full.

Item is the detail data set search item name.
Set is the automatic master data set name.

3.3 Page 5-19

Change: CHAIN HEAD IS FULL
To: FULL CHAIN FOR ***item/set***

Item is the detail data set search item name.
Set is the master data set name

3.4 Page 5-21

Change: NO MANUAL ENTRY FOR DETAIL
To: MISSING CHAIN HEAD FOR ***item/set***

Item is the detail data set search item name.
Set is the master data set name.

HP 32201A EDIT/3000

This article, along with the MIT date coded 1525, comprises the official release of HP 32201A.04.02 EDIT/3000

The following tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32201-1X002 EDITOR MAINTENANCE TAPE

The MIT date coded 1531 contains no updates to EDIT/3000.

The following changes have been made:

1. An error which caused the CHANGE and MODIFY commands to truncate lines longer than 127 characters has been fixed.
2. Several errors concerning line numbering in COBOL format have been fixed. The format of COBOL line numbers is NNN.NNN and the default values for FROM and DELTA are 1.000 and .100, respectively.
3. The KEEP command now allocates all extents of the keep file when no range is specified.
4. All extents of the work file are now allocated when it is opened.
5. An error in the TEXT command which set binary mode when reading from a paper tape reader has been fixed.

HP 30130B 2780/3780 EMULATOR/3000

This article, along with the MIT date coded 1525, comprises the official release of HP 30130B.01.00 2780/3780 EMULATOR/3000.

The following magnetic tapes may be ordered from Corporate Parts Center, Mountain View:

Part No. 30130-1X003 2780/3780 EMULATOR SOURCE
Part No. 3013-1X004 2780/3780 EMULATOR MAINTENANCE

No changes to the documentation are required as a result of this release.

Problems Corrected in 2780/3780 EMULATOR/ 3000.01.00

1. Problem of crashing on termination.
2. Problem with transmission of binary data.
3. Disposition of output files when they are closed has been changed.
4. Input and output procedures will now work.
5. Driver will now properly receive a zero length record terminated with ETB.
6. In Batch mode, if RJE terminates on a line error, a tombstone is printed.

HP 30130B 2780/3780 EMULATOR/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 30130B.01.01 2780/3780 EMULATOR/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 30130-1X004 2780/3780 EMULATOR MAINTENANCE

No changes to documentation are required as a result of this release.

Problems Corrected in 2780/3780 EMULATOR/ 3000.01.01

1. Program file RJE--if started with the system heavily loaded, there was a 50% probability of hanging up the terminal.
2. Driver IOSBSCO
   a. Alleviated the problem that RJE sometimes would not time out in transmit mode when it should.
   b. Previously, if RJLINE was followed by RJIN, it would time out in 3 minutes and give an RJE I/O error 5. Now, if you specify on RJLINE[parameter list];CONNECT=ANSWER, the time out does not occur.

HP 32204A STAR/3000

This article, along with the MIT Date coded 1525, comprises the official release of HP 32204A.00.05 STAR/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32204-1X002 STAR MAINTENANCE

No changes to the documentation are required as a result of this release.

Problems Corrected in STAR/3000.00.05

1. Incorrect calculation of STD.ERRORΔT-DIST. in multiple linear regression procedure.
2. FT.PT.UNDERFLOW in ELEMSTAT procedure. Now STAR/3000 will treat it as zero.
3. Attempted to do SQRT and/or DSQRT of a negative number. Now STAR/3000 will treat these as zero.

The MIT date coded 1531 contains no updates to STAR/3000.

HP 32222A TRACE/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32222A.02.02 TRACE/3000. The MIT date coded 1525 does not contain any updates to TRACE/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32222-1X003 TRACE MAINTENANCE

No changes to documentation are required as a result of this release.

Problems Corrected in TRACE/3000.02.02

1. Coincident stack manipulation by FORTRAN programs during I/O operations and by TRACE as the CONTROL-Y key was hit caused the stack to be in error. The CONTROL-Y key will have no effect as long as the stack is adjusted by I/O operations in a FORTRAN program.
2. The list file has been assigned a file designator, namely, "TRCLIST".

HP 32150A BUILDINT/3000

This article, along with the MIT date coded 1531, comprises the official release of HP 32150.02.03 BUILDINT /3000. The MIT date coded 1525 does not contain any updates to BUILDINT/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 32150-1X002 BUILDINT MAINTENANCE

The declarations for CALENDAR and CLOCK have been added to D00D150A in order for them to be added to the intrinsic file SPLINTR. SPLINTR has been rebuilt.

HP 32102A FORTRAN/3000

This article, along with the MIT date coded 1525, will comprise the official release of HP 32102A.01.05 FORTRAN/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

FORTRAN MAINT. 32102-1X002

No changes to the manual are required.

The following error conditions have been corrected:

1. Equivalence of a byte array with an array with an even number of elements >128 resulted in incorrect computation of the array address.

2. A large (>31 branches) computed GO-TO as the dependent clause of an IF statement generated incorrect code due to the dropping of an indirect cell in the branch table.

3. A trace *GO could, under certain circumstances, cause a bounds violation.

4. Due to the manner in which the Fortran Formatter manipulates the stack, the trace CONTROL-Y feature could get lost. This has been corrected by disabling this trace feature during execution of the formatter.

HP 32211C Compiler Library

Version 03.03 of the Compiler Library is being released on the MIT, Data Coded 1525.

The following changes have been made:

1. A bug in handling zip mode on the series 700 CalComp Plotter has been fixed.

2. The procedures head for FNUM in the SPL intrinsic data file has been corrected so that SPL will properly recognize FNUM as an integer procedure.

A new Compiler Library maintenance tape, Data Code 1525, may be ordered from Corporate Parts Center, Mountain View. The part number is 32211-1X002.

DIAGNOSTICS – HPOFFLN

This article, along with the MIT date coded 1525, comprises the official release of HPOFFLN/3000 1525.

The following magnetic tapes may be ordered from Corporate Parts Center, Mountain View:

Part No. 30000-1X005 S–A DIAGNOSTICS SOURCE
Part No. 30000-1X006 S–A DIAGNOSTICS MAINTENANCE

The following program is affected:

PD211A(update and fix level 01.00)SLEUTH

1. There will be a SLEUTH 3000 Manual update.
2. The 7905 COMMANDS are included.
3. There are 26 new COMMANDS: AR, CHB, CL, CLUB, DISP, FOR, IF, LET, LTIO, NEXT, POLL, RAND, RDA, RQST, RSA, RSYN, RUA, RWO, RWOI, RWU, RWVI, SFM, SKRD, SKWD, VER, VERI. These are fully documented in the manual update.
4. The following modifications were made:

   a. Addition of variables and change of COMMANDS to accept variables.
   b. DUMP will dump variables.
   c. FORMAT will also FORMAT 7905.
   d. ID and IDI work on 7905.
   e. Tape format has changed. The old SLEUTH tapes would not load in BATCH.
   f. PROC now has a parameter for DIRECT I/O interference in PROCEED MODE.
   g. Any COMMAND that states that the LUN must be defined as a moving head disc will now also work on the 7905.
   h. Up to 16 buffers and SIO programs may now be defined (same max length).
   i. Must have 48K of memory to run SLEUTH .01.00.
   j. Have changed the operation of WD for line printers and terminals.

DIAGNOSTICS – HPOFFLN

This article, along with the MIT date coded 1531, comprises the official release of HPOFFLN/3000 1531.

The following magnetic tapes may be ordered from Corporate Parts Center, Moutain View:

Part No. 30000-1X005 S–A DIAGNOSTICS SOURCE
Part No. 30000-1X006 S–A DIAGNOSTICS MAINTENANCE

The following programs are affected:

1. PD319A(update and fix level 01.01)7905 DISC
   a. This is the initial release of PD319A.
2. PD335A(update and fix level 00.02)U.I.INTERFACE
   a. The minimum wait time constant has been changed to 3.5 seconds from 3.99 seconds.
   b. The section select procedure has been corrected to update the internal switch register when bit 0 is enabled.
3. PD336A(update and fix level 00.02)READER/PUNCH INTERFACE
   a. The minimum wait time constant has been changed to 3.5 seconds from 3.99 seconds.
   b. The section select procedure has been corrected to update the internal switch register when bit 0 is enabled.

DIAGNOSTICS – HPONLN

This article, along with the MIT date coded 1531, comprises the official release of HPONLN/3000 1531. The MIT date coded 1525 does not contain any updates to HPONLN/3000.

The following magnetic tape may be ordered from Corporate Parts Center, Mountain View:

Part No. 30000-1X008 ONLINE DIAG MAINTENANCE

The following program is affected:

1. PD366B(update and fix level 01.01)LINE PRINTER
   a. Changes were made to include the HP2617A LINE PRINTER.
   b. The MOD is being updated to reflect these changes.

MIT/Ship Data Schedule

The following is the shipment date schedule for the next three 3000 MITs.. A new six-month schedule will be generated in November.

Date Code	1537	1543	1549
Ship Date	9/12	10/24	12/5

documentation

The following tables list all currently available HP 3000 software manuals. This list supersedes the previous list in the **Communicator.** Copies of manuals and update packages can be obtained from your local Sales and Service Office. The address and telephone number of the office nearest to you are listed in the back of all reference manuals.

Customers in the U.S. may also order directly by mail. Simply list the name and part number of the manuals you need on the Corporate Parts Center form supplied at the back of the **Communicator.** If you require an update package (the items marked N/C in the tables) send your request to:

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

MPE/3000 MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
03000-90096	Multiprogramming Executive General Information Manual	$ 4.00	11/73	
32000-90002	32000C MPE/3000 Reference Manual	19.50	1/75	
32000-90004	32000C MPE/3000 Console Operator's Guide	7.00	1/75	
32000-90006	32000C MPE/3000 System Manager/System Supervisor Manual	13.00	1/75	6/75

LANGUAGE MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
03000-90002	SPL/3000 Reference Manual	$ 7.50	11/73	
03000-90003	SPL/3000 Textbook	25.00	11/73	3/75
	SPL/3000 Textbook Update Package #1	N/C		
03000-90008	BASIC/3000 Interpreter Reference Manual	10.00	7/73	10/74
03000-90014	COBOL/3000 (Version A) Reference Manual	13.00	9/73	11/73
03000-90025	BASIC for Beginners	5.50	11/72	
03000-90047	Cross Assembler for 2100 Computers Reference and Application Manual	17.00	3/75	
03000-90050	HP 32101A BASIC/3000 Interpreter Pocket Guide	2.50	9/74	
32102-90001	FORTRAN/3000 Reference Manual	13.50	6/75	
32103-90001	BASIC/3000 Compiler Reference Manual	3.50	11/74	
32104-90001	RPG/3000 Compiler Reference and Application Manual	22.00	2/75	
32104-90003	RPG Listing Analyzer	0.50	4/75	
32213-90001	COBOL/3000 (Version B) Reference Manual	12.50	10/74	

ADDITIONAL MANUALS

PART NUMBER	MANUAL TITLE	PRICE	DATE	UPDATE
03000-90009	HP 3000 Compiler Library Reference Manual	$10.00	9/73	11/73
03000-90010	HP 3000 Scientific Library Reference Manual	5.00	11/72	
03000-90011	STAR/3000 (Statistical Analysis Routines) Reference Manual	5.50	11/72	
03000-90012	EDIT/3000 Reference Manual*			
03000-90015	HP 3000 Symbol Trace Reference Manual	4.00	2/74	
03000-90019	HP 3000 Computer Systems Reference Manual	14.00	9/73	
03000-90064	FCOPY/3000 Reference Manual	6.00	3/75	6/75
03000-90107	HP 3000 Cross Loader for HP 2100 Computers	11.00	10/74	
03000-90121	A Guide for the Terminal User	7.50	6/75	
30130-90001	2780/3780 Emulator Subsystem Reference and Application Manual	10.00	12/74	
30300-90002	HP 30300A Programmable Controller Reference and Application Manual	12.00	2/75	
32215-90001	IMAGE/3000 Reference Manual	7.00	4/75	
32216-90001	QUERY/3000 Reference Manual	7.00	3/75	
32900-90001	Student Information System Reference Manual	18.00	3/75	
32900-90002	Student Information System – System Overview	7.00	9/74	
32900-90005	Student Information System – Technical Manual	18.50	3/75	
36995-90013	IBM 1130/1800 to HP 3000 FORTRAN Conversion Guide	6.00	2/75	5/75
32214-90001	32214B Sort/3000 Reference Manual	6.50	4/75	

*Price to be determined. Will be included in next issue of the **Communicator.**

training schedule

The schedule for software training courses related to the HP 3000 is provided below. Included are courses scheduled for the next four months (September through December).

Each issue of the **Communicator** will provide timely information on training to assist you in registering for classes applicable to the operation of your system.

SOFTWARE COURSES

COURSE NUMBER	PRODUCT	COURSE LENGTH	DATES	LOCATION
22962A	3000 Commercial/Business User	5 days	Sept 8–12	Cupertino
			Oct 20–24	Rockville
			Nov 3–7	Cupertino
			Dec 1–5	Rockville
22963A	3000 Scientific/Engineering User	5 days	Sept 22–26	Rockville
			Oct 6–10	Cupertino
			Nov 17–21	Rockville
			Dec 8–12	Cupertino
22964A	3000 System Management	3 days	Sept 15–17	Cupertino
			Sept 29–Oct 1	Rockville
			Oct 13–15	Cupertino
			Oct 27–29	Rockville
			Nov 10–12	Cupertino
			Nov 24–26	Rockville
			Dec 8–10	Rockville
			Dec 15–17	Cupertino
22956A	3000 Image	5 days	Sept 22–26	Cupertino
			Nov 17–21	Cupertino
22974A	Minicomputers in Manufacturing	2 days	Oct 16–17	Cupertino

HP Training Centers

Training is conducted in the U.S.A. at facilities in Cupertino, California and Rockville, Maryland.

Each Training Center is staffed with professional instructors. Courses are designed such that the student will receive both classroom instruction and practical, hands-on experience. By attending the courses in the recommended sequence for your particular HP system, the student will gain the most beneficial training available to meet the needs of your specific application.

Registration

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

Accommodations

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

Cancellations

In the event you find you cannot attend a particular class we would appreciate your prompt notification.

Eastern Training Center

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

Western Training Center

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

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

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

direct mail order form

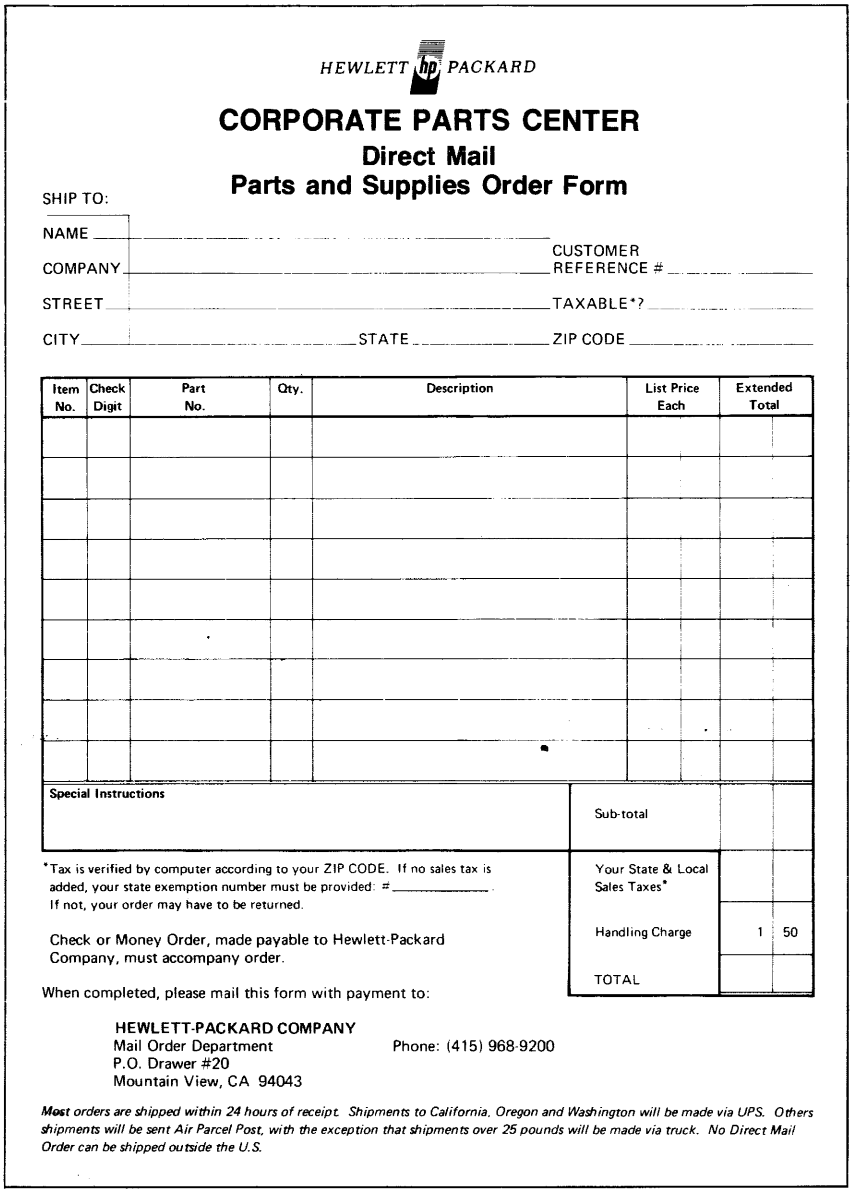
To expedite your order for software, manuals, or other materials described in this publication, use this form to order directly from the Corporate Parts Center in Mountain View, California.

1. Enter your name, address, customer reference number, and tax exemption information.
2. List the item or items you want by part number and description.
3. Compute the amount due and enclose a check or money order payable to Hewlett-Packard.

If you need assistance in placing your order, contact your local HP Sales Office.

Note: **Update packages** and **manual supplements** cannot be ordered from the Corporate Parts Center. Direct your request to:

Software Publications Distribution
Hewlett-Packard
11000 Wolfe Road
Cupertino, Ca. 95014



HEWLETT 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. Others 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.

Computer Systems COMMUNICATOR
Published by: HP Data Systems Division
Customer Engineering Department
11000 Wolfe Road
Cupertino, Ca. 95014