Worldwide Response Center

HP 3000 APPLICATION NOTE #96

Tape Labels

Unlimited

HEWLETT PACKARD

May 1993
Document P/N 5961-9647

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

RESPONSE CENTER APPLICATION NOTES

HP 3000 APPLICATION NOTES *are published by the Worldwide Response Center and are distributed with the Software Status Bulletin. These notes address topics where the volume of calls received at the Center indicates a need for addition to or consolidation of information available through HP support services.*

Following this publication you will find a list of previously published notes and a Reader Comment Sheet. You may use the Reader Comment Sheet to comment on the note, suggest improvements or future topics, or to order back issues. We encourage you to return this form; we'd like to hear from you.

HP 3000 Application Note # 96

Tape Labels Unlimited

Introduction

This Application Note's primary purpose is to describe the steps necessary to construct an IBM or ANSI tape label manually, bypassing MPE's labelled tape subsystem, which has the advantage that you have much greater control over the information that can be included on the label. Additionally, a technique for mixing block sizes on tape using NOBUF i/o and undefined record sizes is explained, along with the use of FCOPY parameters SKIPEOF, TO=* and EBCDICOUT.

See also HP 3000 Application Note 86 for answers to other questions on the reading and writing of IBM labelled tapes

If you want to create an IBM labelled tape to send to another site, then this can be achieved through HP's labelled tape facility. Labels can be written to tape in ANSI or IBM format, by specifying a LABEL parameter on the file equation to the tape device. The default is ANSI, but use of the IBM option, will cause the label to be written as per IBM's documentation of their format.

The problem with this method is that certain option fields on the label cannot be completed, and also that the field DATA SET IDENTIFIER is restricted to the possible values of the formal file designator (in other words the "T" from FILE T;DEV=TAPE ...). The workaround is to build the label manually, and then you have complete control over its contents.

1. The Theory

The label on a labelled tape is actually a set of files containing records in a predefined layout which precede and follow the data files. The records on the tape follow the sequence below where the VOL1, HDR1, HDR2, EOF1 and EOF2 type records have a precisely defined format which differs between the IBM and ANSI standards, although the tape layout is the same:

```
<beginning of tape mark>
VOL1
HDR1     <---------------- 1st file on tape (part of 1st data file's
HDR2                       label)
<end of file mark>
data
data
....     <---------------- 2nd tape file (1st data file)
data
<end of file mark>
EOF1     <---------------- 3rd tape file (part of 1st data file's
EOF2                       label)
<end of file mark>
HDR1     <---------------- 4th tape file (part of 2nd data file's
HDR2                       label)
<end of file mark>
data
data     <---------------- 5th tape file (2nd data file)
....
data
<end of file mark>
EOF1
EOF2
<end of file mark>
  .....
         <---------------- and so on, until after the last file.
EOF1
EOF2
<end of file mark>
<end of file mark>  (2 end of file marks signify the end of the
                     written portion of the tape)
```

From the above, it can be seen that the labels are simply files interspersed with the data, so that if you know the format, you can build them and write them to the tape yourself with FCOPY (or a program).

Each data file has its own header and trailer files which precede and follow it. The header file contains HDR1 and HDR2 records, and the trailer file contains EOF1 and EOF2 records. In the case of the very first header file on a tape reel, there will be a special VOLn record as well.

The following examples show how the specific VOL, HDR and EOF records can be constructed.

```
FILENAME  CODE  ------------LOGICAL RECORD-----------
                  SIZE  TYP        EOF      LIMIT R/B
IBMLAB11           80B  FA           3          3   1
IBMLAB12           80B  FA           2          2   1
```

IBMLAB11 is a typical IBM header file, and in this case is the first header file on the tape, so contains 3 records, as follows:

```
VOL1CA90020
HDR1CCA.ZF4.PLAN0123 00000100010001000100 91067 000000000001HP MPE 3000
HDR2F0616000080 0                      B
```

IBMLAB12 is the trailer file, containing 2 very similar records:

```
EOF1CCA.ZF4.PLAN0123 00000100010001000100 91067 000000000001HP MPE 3000
EOF2F0616000080 0                      B
```

a. VOLn record

The first record of the first label on a tape is always prefixed VOLn, where 'n' is the number of the reel in the labelled tape set.

Bytes	Contents
1/4	"VOLn"
5/10	Volume ID (in this example, "CA9002")
11	Accessibility - "0" if IBM, else " "
12/41	Unused (blanks)
42/51	Tape Owner (IBM only and optional) or blanks
52/79	Unused (blanks)
80	Label-standard version - "1" if ANSI, else " "

b. HDR1 record

Bytes	Contents
1/4	"HDR1"
5/21	File name or "data set identifier" (IBM terminology) - in this example CCA.ZF4.PLAN0123 with one trailing blank. MPE's labelled tape facility only writes and reads the first 8 bytes and gets this information from the formal file designator (e.g. the "T" in FILE T;DEV=TAPE)

22/27	Which reel of the set we start on (in this case 1 - so "000001")
28/31	Counts the reels which contain this file - so we start with "0001", but if the file continued onto another reel, the HDR1 record for the continued file should contain "0002" in these bytes.
32/35	Counts the files in the labelled tape set, so in this example "0001" for the first file. The next file's HDR1 record would contain "0002", and so on.
36/39	IBM Generation number (MPE writes "0001") or blanks (ANSI)
40/41	IBM Version number (MPE writes "00"), or blanks (ANSI)
42/47	File creation year, and day in year (e.g. " 91067")
48/53	File expiration year and day (e.g. " 00000" means that the tape is always "expired", and " 93001" means the tape expires on 1st Jan 1993).
54	"0" if IBM. %230 or blank if ANSI (%230 indicates that the file is lockworded - see HDR2 bytes 16/23)
55/60	IBM only - number of blocks (MPE always writes "000000") or blanks (ANSI).
61/73	System ID. MPE will always create this as "HP MPE 3000" but you can specify the actual source machine instead you prefer.
74/80	Blanks.

c. HDR2 record - IBM format

Bytes	Contents
1/4	"HDR2"
5	File record format (in this case "F" for fixed, but the use of undefined/variable length records is not advised since different operating systems use different size counters for the actual record lengths, so they're usually incompatible).
6/10	Block length in bytes (in this case "06160")
11/15	Record length in bytes (in this case "00080")

16	Unused (leave it blank)
17	IBM position - "0" no volume switch, "1" switch occurred (in this case, "0")
18/38	Blanks (unused)
39	IBM block attribute: "B" for blocked records, "S" for spanned, "R" for blocked and spanned, blank for neither.
40/80	Blanks (unused)

d. HDR2 record - ANSI format

Bytes	Contents
1/4	"HDR2"
5	File record format - "F" fixed, "V" variable, "U" undefined. All others are treated as undefined
6/10	Block length in bytes
11/15	Record length in bytes
16/23	MPE file lockword - see HDR1 byte 54
24/36	Blanks (unused)
37	Record type - "A" ascii, "B" binary
38	Carriage control - "C" control, " " no control
39/80	Blanks (unused)

e. EOF1 record

Bytes	Contents
1/4	"EOF1"
5/54	Same as HDR1
55/60	Number of data blocks since the beginning of the file (i.e. data file block count). In this example, "000001"
61/80	Same as HDR1

f. EOF2 record

Bytes	Contents
1/4	"EOF2"
5/80	Same as HDR2

2. The Technical Details

Having determined the layout of the header and trailer label files, the next problem is to copy the data and label information to the tape. This has to take place, such that the label files are written 1 record per block, with block/record length at 80 bytes, yet the actual data files may have different record/blocking characteristics (almost certainly - since larger blocks use up the tape more efficiently than small ones!).

There are a few features of MPE and FCOPY that have to be used to achieve this, so they are explained below, before the examples where they are demonstrated.

a. NOBUF I/O

In order to mix the block sizes on the tape, we first need to understand that as far as the tape is concerned, we are writing to it one block at a time. Each block may contain many logical records, but this split is done when the data is written to or is read from the tape, and is controlled by the way we file equate to it in MPE. For example, we could have data written to the tape in blocks of 4000 bytes.

```
:BUILD DATAFILE;REC=-400,1,F,ASCII;DISC=20000
:FILE T;DEV=TAPE;REC=-80,50,F,ASCII
:FCOPY FROM=*T;TO=DATAFILE

:BUILD DATAFILE;REC=-400,1,F,ASCII;DISC=10000
:FILE T;DEV=TAPE;REC=-400,10,F,ASCII
:FCOPY FROM=*T;TO=DATAFILE
```

The first example reads the blocks, but splits them into records of length 80 bytes each, whereas the second will read the same data, but splits into 400 byte records. In the first example, the destination disk file actually has 400 byte records, but bytes 81 through 400 will be unused and padded with blanks.

When we want to create a labelled tape using FCOPY to write the label files, instead of specifying a record length and blocking factor the usual way, we instead say that the tape has UNDEFINED length records, 1 record per block, and a record size which is as large as the largest block which will be needed (according to the data files we need to put onto the tape). So, for example, if we have 3 data files to write which have a record length of 200 bytes and blocking factor of 10, and another data file with record length 256 bytes and blocking factor of 16, we multiply to get the block sizes (2000 and 4096) and see that 4096 bytes will be our biggest block.

Our file equation to tape will then be:

```
:FILE T;DEV=TAPE;REC=-4096,1,U,ASCII
```

Next, we take advantage of the situation that with undefined-length output records, the actual record size written will be equal to the length of the data record supplied by the FCOPY. So, the files containing our label information need to be kept on disk very carefully with 80 byte records (the blocking factor does not matter). This will result in 80 byte records, 1 record per block being written to the tape.

Finally, in order to get the data files blocked correctly on the tape, we first block them correctly on the disk:

```
FILENAME  CODE  ------------LOGICAL RECORD-----------
                   SIZE  TYP        EOF       LIMIT R/B

DATA1              200B  FA         1057        2000  10
DATA2              200B  FA        20398       25000  10
DATA3              200B  FA          260        1000  10
DATA4              256B  FA         4001        5000  16
```

Then, we file equate them with the ;NOBUF option, so that we read the files from disk, one block at a time instead of one record at a time ...

```
FILE IN1=DATA1;NOBUF
FILE IN2=DATA2;NOBUF
FILE IN3=DATA3;NOBUF
FILE IN4=DATA4;NOBUF
```

Using FCOPY FROM=*IN1;TO=*T will now supply physical blocks of 2000 bytes (equivalent to 10 logical records of 200 bytes each) in each write to the tape.

b. FCOPY ... ;TO=*

We need to be able to write each file to the tape without rewinding it in between files. We achieve this via the * option, which tells FCOPY to use the file that you were last writing to (or reading from), without closing it. This option is only available at FCOPY's > prompt:

```
:FCOPY
>FROM=IBMLAB11;TO=*T;EBCDICOUT
>FROM=DATA1;TO=*;EBCDICOUT;SKIPEOF=,+1
>FROM=IBMLAB12;TO=*;EBCDICOUT;SKIPEOF=,+1
>...
```

c. FCOPY ... ;SKIPEOF=

We need to be able to write an End Of File (EOF) mark in between each file - just using the TO=* will not do this for us, it only appends the output from the next input file onto the end of the last one. The SKIPEOF parameter in FCOPY instructs FCOPY to skip the specified count of EOF marks in the fromfile (before the comma) or the tofile (after the comma). The plus or minus ("+" or "-") indicates that this is relative to the current file position, rather than absolute, so the syntax below will direct FCOPY to skip one EOF in the tofile before writing the next data records.

```
>...;SKIPEOF=,+1
```

Since there will not be an EOF on the tape (we are writing to it, so what is already there is irrelevant), this directive forces FCOPY to write an EOF for us, so that we can skip it.

d. FCOPY ... ;EBCDICOUT

If the tape label we are writing is ANSI, and intended for another HP3000, or a machine which reads ASCII rather than EBCDIC, then this option will not be required. If the tape is intended for a machine which will use EBCDIC rather than ASCII for its character representation, then we need to convert both the label information, and the data. This is done via the FCOPY EBCDICOUT option. There is also an equivalent EBCDICIN option which would be used to read tapes created in EBCDIC.

Note Unfortunately, there is a problem with the use of EBCDICOUT to tape files when the records are defined as type U - undefined, or V - variable. This is documented in SR 5000461251. There are two feasible workarounds - one is to specify the tape file as BINARY instead of ASCII in the output file equation, and the second is to convert the data and label files to EBCDIC on disk first, before copying them to tape.

3. Example 1 - IBM Labelled Tape Containing a Single File

This is the simplest situation, and can easily be converted to the ANSI label requirement by omitting the EBCDICOUT option and structuring the label files in ANSI format instead of IBM.

Step 1:

Consider the blocking factor and record size requirements of the file to be placed on tape, and block the file on disk accordingly. In this example, the recipient of the tape requires the tape to be constructed with a block size of 5000 bytes, with records of length 100 bytes.

Step 2:

Consider the label information - you will need to know the volume id, the tape owner (optional), the filename (or "data set identifier"). Optionally, you may also be asked to supply a specific source system ID, or even have the tape label layout specified explicitly by your intended recipient. You also need to know how many records there will be in the source file, so you can compute the number of blocks that will be written to the tape. In the example, the number of records is 1143. Divide this by the blocking factor (50) to get the number of blocks to go to tape - 23.

Step 3:

Construct the label files:

```
:BUILD HEAD1;REC=-80,1,F,ASCII;DISC=3
:BUILD TRAIL1;REC=-80,1,F,ASCII;DISC=2
:FCOPY FROM=;TO=HEAD1
HP32212A.05.00 FILE COPIER (C) HEWLETT-PACKARD CO. 1990
VOL1EXAMP10
HDR1DATA1.EXAMP1     00000100010001000100 92023 000000000000RC1.HP3000
HDR2F0500000100 0                         B
:
EOF FOUND IN FROMFILE AFTER RECORD 2
3 RECORDS PROCESSED *** 0 ERRORS

END OF SUBSYSTEM
:FCOPY FROM=;TO=TRAIL1
HP32212A.05.00 FILE COPIER (C) HEWLETT-PACKARD CO. 1990
EOF1DATA1.EXAMP1     00000100010001000100 92023 000000000023RC1.HP3000
EOF2F0500000100 0                         B
:
EOF FOUND IN FROMFILE AFTER RECORD 1
2 RECORDS PROCESSED *** 0 ERRORS

END OF SUBSYSTEM
:
```

Note In the above example, we have used FCOPY from the screen (FROM= blanks) to enter the data directly. Press (Return) after each line, then type ":" and press (Return) to indicate to FCOPY that you've reached the end of the input file.

Step 4:

Block the data file on disk the way you want it on tape, that is 50 records per block (block size = 5000).

```
:BUILD NEWDATA;REC=-100,50,F,ASCII;DISC=1143
:FCOPY FROM=DATA1;TO=NEWDATA
HP32212A.05.00 FILE COPIER (C) HEWLETT-PACKARD CO. 1990
EOF FOUND IN FROMFILE AFTER RECORD 1142
1143 RECORDS PROCESSED *** 0 ERRORS
END OF SUBSYSTEM
:LISTF NEWDATA,2

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

NEWDATA           100B  FA        1143       1143  50      448  2  *
```

Step 5:

Copy the data, with its label files to tape:

```
!JOB MAKETAPE,USER.ACCOUNT
!FILE T;DEV=TAPE;REC=-5000,1,U,BINARY
!FILE DATAFILE=NEWDATA;NOBUF
!FCOPY
FROM=HEAD1;TO=*T;EBCDICOUT
FROM=*DATAFILE;TO=*;EBCDICOUT;SKIPEOF=,+1
FROM=TRAIL1;TO=*;EBCDICOUT;SKIPEOF=,+1
EXIT
!EOJ
```

4. Example 2 - IBM Labelled Tape Containing Two Files

This example shows you how to construct the second (and therefore third, fourth etc ...) file header and trailer label files, in order to write multiple files to the same tape reel.

Step 1:

Consider the blocking factor and record size requirements of the files to be placed on tape, and block the files on disk accordingly. In the example, the recipient of the tape requires the first file to have a block size of 4096 bytes, with records of length 128 bytes, and the second file to have block size 500 bytes, with records of length 50 bytes. Note that the largest block will be 4096 bytes, so this is the length of the record which is declared in the file equation to the tape.

Step 2:

Consider the label information - you will need to know the volume id, the tape owner (optional), and for each file, the filename (or "data set identifier"). set identifier"). Optionally, you may also be asked to supply a specific source system ID, or even have the tape label layout specified explicitly by your intended recipient. You also need to know how many records there will be in each source file, so you can compute the number of blocks that will be written to the tape for each. In this example, the first file contains 16789 records - at 32 records per block, this is 525 blocks. The second file has 9 records - at 10 records per block this will be only 1 block.

Step 3:

Construct the label files (see EXAMPLE 1 for FCOPY method of doing this). The 4 files, HEAD1, TRAIL1, HEAD2 and TRAIL2 should look as follows:

FILENAME	CODE	LOGICAL RECORD				
		SIZE	TYP	EOF	LIMIT	R/B
HEAD1		80B	FA	3	3	1
HEAD2		80B	FA	2	2	1
TRAIL1		80B	FA	2	2	1
TRAIL2		80B	FA	2	2	1

```
HEAD1:

VOL1EXAMP20
HDR1FILE1.EXAMPLE2   0000010001000100 92023 923650000000HP3000 RC6
HDR2F0409600128 0                     B

TRAIL1:

EOF1FILE1.EXAMPLE2   0000010001000100 92023 923650000023HP3000 RC6
EOF2F0409600128 0                     B
```

```
HEAD2:

HDR1FILE2.EXAMPLE2   0000010001000200100 92023 923650000000HP3000 RC6
HDR2F0050000050 0                        B

TRAIL2:

EOF1FILE2.EXAMPLE2   0000010001000200100 92023 923650000001HP3000 RC6
EOF2F0050000050 0                        B
```

Step 4:

Block the data files on disk the way you want them on tape (see EXAMPLE 1 for how to do this). Your files should look as follows:

```
FILENAME  CODE  ------------LOGICAL RECORD-----------
                    SIZE  TYP        EOF      LIMIT R/B

FILE1               128B  FA       16789      16789  32
FILE2                50B  FA           9          9  10
```

Step 5:

Copy the data, with its label files to tape:

```
!JOB IBMTAPE,USER.ACCOUNT
!FILE T;DEV=TAPE;REC=-4096,1,U,BINARY
!FILE DATAFIL1=FILE1;NOBUF
!FILE DATAFIL2=FILE2;NOBUF
!FCOPY
FROM=HEAD1;TO=*T;EBCDICOUT
FROM=*DATAFIL1;TO=*;EBCDICOUT;SKIPEOF=,+1
FROM=TRAIL1;TO=*;EBCDICOUT;SKIPEOF=,+1
FROM=HEAD2;TO=*;EBCDICOUT;SKIPEOF=,+1
FROM=*DATAFIL2;TO=*;EBCDICOUT;SKIPEOF=,+1
FROM=TRAIL2;TO=*;EBCDICOUT;SKIPEOF=,+1
EXIT
!EOJ
```

5. Example 3 - ANSI Labelled Tape Containing Two Files

This example is identical to EXAMPLE 2, except that the header and trailer label files, and the sample batch job to create the tape is done in ANSI format:

The four label files (ANSIHD1, ANSITRL1, ANSIHD2 and ANSITRL2) should look as follows:

```
FILENAME  CODE  ------------LOGICAL RECORD-----------
                  SIZE  TYP         EOF       LIMIT R/B

ANSIHD1            80B  FA            3           3   1
ANSIHD2            80B  FA            2           2   1
ANSITRL1           80B  FA            2           2   1
ANSITRL2           88B  FA            2           2   1

ANSIHD1:

VOL1EXAMP2
HDR1FILE1          00000100010001       92023 92365       HP MPE 3000
HDR2F0409600128                    A
```

Note Although you cannot see it here, column 80 of the VOL1 line contains a "1" to indicate that this is an ANSI labelled tape.

```
ANSITRL1:

EOF1FILE1          00000100010001       92023 923650000023HP MPE 3000
EOF2F0409600128                    A

ANSIHD2:

HDR1FILE2          00000100010002       92023 92365       HP MPE 3000
HDR2F0050000050                    A

ANSITRL2:

EOF1FILE2          00000100010002       92023 923650000001HP MPE 3000
EOF2F0050000050                    A
```

The job to create the tape should look as follows:

```
!JOB ANSITAPE,USER.ACCOUNT
!FILE T;DEV=TAPE;REC=-4096,1,U,BINARY
!FILE DATAFIL1=FILE1;NOBUF
!FILE DATAFIL2=FILE2;NOBUF
!FCOPY
FROM=ANSIHD1;TO=*T
FROM=*DATAFIL1;TO=*;SKIPEOF=,+1
FROM=ANSITRL1;TO=*;SKIPEOF=,+1
FROM=ANSIHD2;TO=*;SKIPEOF=,+1
FROM=*DATAFIL2;TO=*;SKIPEOF=,+1
FROM=ANSITRL2;TO=*;SKIPEOF=,+1
EXIT
!EOJ
```

6. Multi-Reel Labelled Tape Sets

It is possible that the files you want to copy to labelled tapes will require more than one reel. This presents some problems, because when the physical end of tape mark is detected by FCOPY, it will terminate, instead of asking you to mount the next reel (as it would if you were writing to the tapes using MPE's labelled tape facility instead of constructing the labels manually).

The author has not attempted to create multi-reel labelled tape sets manually, but expects that the easiest way to do this would be to calculate where the reel split is going to occur, then allowing a reasonable margin of error, use the FCOPY SUBSET option to restrict the number of records/blocks being written to the current reel before the end of tape mark is found.

With multi-reel tapes, you will need to write a VOLn label at the start of each new reel, take great care with the first HDR1 and HDR2 labels if a file has been split across the reels, and write EOV1 and EOV2 labels at the end of the preceding reel instead of EOF1 and EOF2 (same format, different prefix).

Published Application Notes

HP 3000

Following is a list of the Application Notes published to date. If you would like to order single copies of back issues please use the *Request Form* attached and indicate the number(s) of the note(s) you need, and the part number(s).

Note #	Part Number	Topic
1	5958-5824	Printer Configuration Guide - Version 1
2	5960-2841	Terminal types for HP 3000 HPIB Computers - Version 1
3	5960-2842	Plotter Configuration Guide
4	5960-2843	Printer Configuration Guide - Version 2
5	5960-2844	MPE System Logfile Record Formats
6	5960-2845	Stack Operation
7	5960-2846	COBOL II/3000 Programs: Tracing Illegal Data
8	5960-2847	KSAM Topics: COBOL's Index I/O: File Data Integrity
9	5960-2848	Port Failures, Terminal Hangs, TERMDSM
10	5960-2849	Serial Printers - Configuration, Cabling, Muxes
11	5960-2850	System Configuration or System Table Related Errors
12	5960-2851	Pascal 3000 - Using Dynamic Variables
13	5960-2852	Terminal Types for HP 3000 HPIB Computers - Version 2
14	5960-2853	Laser Printers - A Software and Hardware Overview
15	5960-2854	FORTRAN Language Considerations - A Guide to Common Problems
16	5960-2855	IMAGE: Updating to TurboIMAGE & Improving Database Loads
17	5960-2856	Optimizing VPLUS Utilization
18	5960-2857	The Case of the Suspect Track for 792X Disc Drives
19	5960-2858	Stack Overflows: Causes & Cures for COBOL II Programs
20	5960-2859	Output Spooling
21	5960-2860	COBOLII and MPE Intrinsics
22	5960-2861	Asynchronous Modems

Note #	Part Number	Topic
23	5960-2862	VFC Files
24	5960-2863	Private Volumes
25	5960-2864	TurboIMAGE: Transaction Logging
26	5960-2865	HP 2680A, 2688A Error Trailers
27	5960-2866	HP Trend: An Installation and Problem Solving Guide
28	5960-2867	The Startup State Configurator
29	5960-2868	A Programmer's Guide to VPLUS 3000
30	5960-2869	Disc Cache
31	5960-2870	Calling the CREATEPROCESS Intrinsic
32	5960-2871	Configuring Terminal Buffers
33	5960-2872	Printer Configuration Guide - Version 3
34A	5960-2873	RIN Management (Using COBOLII Examples) (A)
34B	5960-2874	Process Handling (Using COBOLII Examples) (B)
35	5960-2875	HPDESK IV (Script files, FSC, and Installation Considerations)
34C	5960-2876	Extra Data Segments (Using COBOLII Examples) (C)
36	5960-2877	Tips for the DESK IV Administrators
37	5960-2878	AUTOINST: Trouble-free Updates
38	5960-2879	Store/Restore Errors
39	5960-2880	MRJE Emulates a HASP Workstation
40	5960-2881	HP 250 / 260 to HP 3000 Communications Guidelines
41	5960-2882	MPE File Label Revealed
42	5960-2883	System Interrupts
43	5960-2884	Run Time Aborts
44	5960-2885	HPPA Patching Conventions for HP3000 900 Series Processors - Version 1
45	5960-2886	Vplus & Multiplexers
46	5960-2887	Setting Up an HPDesk HPTelex for the First Time
47	5960-2900	Customizing Database Data Items & Changing Passwords in JCL Files
48	5959-9215	Printer Configuration - Version 4
49	5959-9227	Configuring DATACOMM Products Into MPE
50	5959-9228	VFC's for Serial Printers

Note #	Part Number	Topic
51	5959-9237	Terminal Types for the HP 3000 HPIB Computers
52	5959-9242	Configuring MRJE
53	5959-9245	Using Special Characters on the 700/9x Series Terminals
54	5959-9251	Improving Database Performance
55	5959-9258	Customized Message Catalogs and Help Facilities
56	5959-9266	BRW Tips for Beginners
57	5959-9270	Configuring the HP 2334A Plus & HP 2335A As a Statistical Multiplexer
58	5959-9274	HPPA Pathing Conventions for HP3000 900 Series Processors - Version 2
59	5959-9289	HP 2334A and HP 2334A Configuration Recipes
60	5959-9301	TurboIMAGE's I-FILES and J-FILES
61	5959-7385	HPDeskManager - Looking Behind the Scenes
62	5959-7803	Setting Up a System Dictionary
63	5959-7834	Configuring Telesupport Modems for MPE V/E Systems
64	5960-1816	Finding Solutions in HP SupportLine
65	5960-1817	Using the Electronic Call Feature of HP SupportLine
66	5960-1818	Using the Feedback Feature of HP SupportLine
67	5960-1819	Printing Documents from HP SupportLine
68	5960-1820	HP SupportLine Commands
69	5960-2901	Nonsystem Volume Sets and the Migration of Private Volumes to an S9000 HP 3000
70	5960-2907	Modem Links for Remote Console and Standard DTC Connections on Commercial XL HPPA Systems
71	5960-2918	Asynchronous Cabling
72	5960-2919	BRW Tips and Tricks
73	5960-2998	SNA NRJE Configuration
74	5960-2999	SNA IMF Configuration
75	5060-3000	XL NRJE Configuration

Note #	Part Number	Topic
76	5960-4301	XL IMF Configuration
77	5960-4302	Calling the BRW Intrinsics
78	5960-4303	PUB.SYS What Is Behind It?
79	5960-4625	Conquest of Disc Space
80	5960-4633	Looking Behind the Scenes of Resource Sharing
81	5960-4637	MPE/XL System Interrupt Recovery Procedures
82	5960-4347	Private Volumes
83	5960-4396	Serial Printer Configuration
84	5960-4334	How to Migrate FORTRAN Programs to Newer Compilers and XL Hardware
85	5960-4335	The Optimization of Programs in MPE/XL
86	5960-4643	IBM Labeled Tapes Questions and Answers
87	5960-4666	Image Logging for HP Financial Accounting Databases
88	5960-4672	Native Mode Spooler Questions and Answers
89	5960-4673	AUTOINST/XL Questions and Answers
90	5960-4701	The New Spooler
91	5960-6659	Using the Port Structure Under MPE/XL
92	5960-6696	SUBNET 3000
93	5960-6697	Native Mode Spooler Questions and Answers Version 2
94	5960-8223	RPG/XL Intrisic Interface
95	5961-1689	LaserRx Questions and Answers
96	5961-9647	Tape Labels Unlimited

NOTES

NOTES

NOTES

NOTES

HP 3000 Application Note Request

☐ Please send me the HP 3000 Application Notes listed below. To order an Application Note just fill in the Application Note number and the part no. in the space provided below. (Use the Published Application Notes form)

AN ___ 59 ___-___ AN ___ 59 ___-___

AN ___ 59 ___-___ AN ___ 59 ___-___

AN ___ 59 ___-___ AN ___ 59 ___-___

AN ___ 59 ___-___ AN ___ 59 ___-___

AN ___ 59 ___-___ AN ___ 59 ___-___

Business phone Extension Best time to call

Name Title/Dept.

Company Division

Mailing Address Mail Stop/Bldg./Rm.

City State Zip Code

3515

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS MAIL PERMIT NO. 609 CUPERTINO, CA

POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT-PACKARD
19310 PRUNERIDGE AVE
BLDG 49 A M
CUPERTINO CA 95014-9826

Customer Comment Card

Application Note No.: _____

Part No.: _____

We welcome your evaluation of this Application Note. Your comments and suggestions will help us improve our publications. Attach additional pages if necessary.

Please circle the following Yes or No:

■ Is the information technically accurate?	Yes	No
■ Are instructions complete?	Yes	No
■ Are concepts and wording easy to understand?	Yes	No
■ Are the examples and pictures helpful?	Yes	No
■ Is the format of this note convenient in size, arrangement and readability?	Yes	No
■ Did you receive the published application notes requested in a timely manner?	Yes	No

Additional Comments and/or suggestions for future application notes: _____

_____

_____

_____

_____

Please provide:

Name: _____ Title: _____

Company: _____ Address: _____

City: _____ State: _____ Zip Code/Country _____

Please send to :

HEWLETT PACKARD

No postage is required. Just remove this card, fold so that the pre-addressed label is on the outside, secure and mail.

Thank you for your assistance.

Printed in U.S.A. 1990

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 95, MT. VIEW, CA

POSTAGE WILL BE PAID BY ADDRESSEE

Learning Products Manager
Hewlett-Packard Company
100 Mayfield Avenue
Mail Stop 37MA
Attention: George Enos
Mt. View, CA 94043

FOLD FOLD

** For HP Internal Reference Only **

Manufacturing Part Number

5961-9647

Customer Order Number

NONE

Printed in USA