HEWLETT hp PACKARD

USING FILES

a guide for new users
of HP 3000
computer systems

FILES

using files

a guide for new users
of HP 3000 computer systems

First Edition. . . . . July 1977
Second Edition . . . . April 1978

HEWLETT PACKARD

5303 Stevens Creek Blvd., Santa Clara, California 95050

Part No. 30000-90102
Product No. 32002B

Printed in U.S.A. 4/78

preface

This guide introduces you to methods for manipulating files on HP 3000 computer systems. It does not tell you everything you need to know about accessing files with your own programs or discuss disc file structure in detail but rather shows you which commands, subsystems, and utility programs can help you manipulate files and defines some basic terms relating to files.

The guide assumes you have had some experience with data processing methods and, specifically, with files. If you don't know how to log on to MPE (HP 3000's operating system) and use MPE commands, read Section I of *Using the HP 3000* before reading further in this guide.

If you plan to use an IMAGE data base, you may find some parts of this guide helpful but the *IMAGE Data Base Management Reference Manual* contains most of the information you will need.

For instructions on using files to compile and prepare programs, read *Using the HP 3000* and the appropriate programming language reference manual as well as this guide.

A glossary is provided at the end of the guide - look there if you encounter unfamiliar terms while reading the guide.

Part numbers for the manuals referenced are listed in Appendix A.

contents

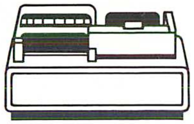
3000 system files

3000 system files

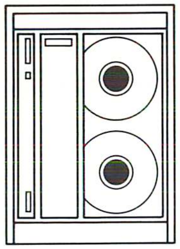
Most people who have used computers think of a file as data stored on either a magnetic tape or a disc.

As you begin using your 3000 system, you will find that MPE treats information being entered from any input device

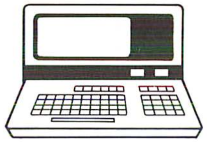
Card Reader



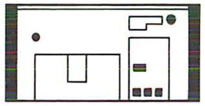
Magnetic Tape



Terminal

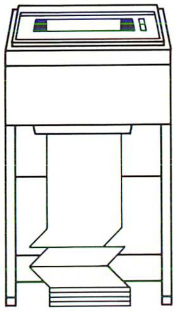


Paper Tape Reader

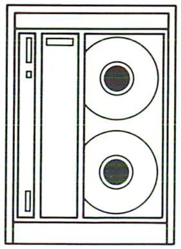


or written to any output device

Line Printer

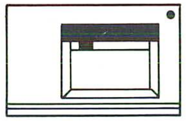


Magnetic Tape



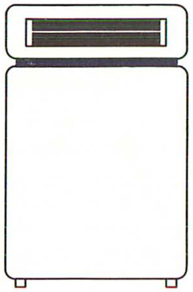
Terminal

Paper Tape Punch Unit



or stored on a disc

Disc



as a file.

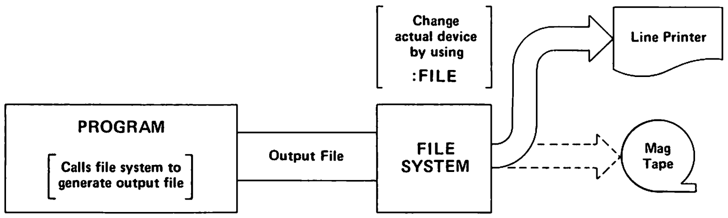
THE FILE SYSTEM

All access to files is accomplished through a portion of the HP 3000 Multiprogramming Executive operating system called the File System. Since the File System manages all input and output through files, you can access very different devices in a standard, consistent way. Access to data by a program may be

$\boxed{\text{device-independent}}$.

This means each time you run a program, you can tell MPE where the file is and what its characteristics are by using a :FILE command.

Suppose a program writes information to a file it calls OUTFILE. The data may be written to magnetic tape during one execution of the program and to a line printer during another execution.


Change actual device by using :FILE
Line Printer
PROGRAM
Calls file system to generate output file
Output File
FILE SYSTEM
Mag Tape

The major characteristics of the device and type of access must also be considered; for example, a program cannot write to a card reader.

The name by which the program refers to the file is its

$\boxed{\text{formal file designator}}$.

The name by which MPE knows the file is its

$\boxed{\text{actual file designator}}$.

For example, a program may refer to an output file as OUT123 (or a FORTRAN program may refer to an output file as FTN10). Before you run the program you can use a :FILE command to equate the formal file designator (OUT123) to a particular device type:

```
:FILE OUT123; DEV=TAPE  or  :FILE FTN10; DEV=TAPE
:FILE OUT123; DEV=LP    or  :FILE FTN10; DEV=LP
```

If you want to write output to a disc file, you can use the :FILE command to equate the formal file designator to an actual file designator (the name of an existing permanent or temporary disc file, or a new disc file). For example, to equate it to an existing permanent disc file named FILEXYZ you would use this command:

```
:FILE OUT123=FILEXYZ
```

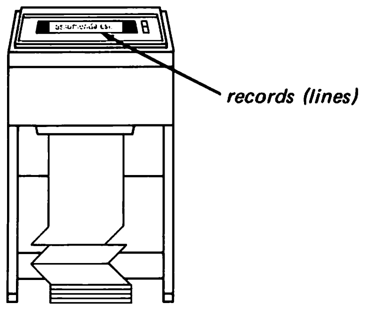
Unless you specify a device type, the default device type is DISC.

DEVICES

When the system supervisor configures the system, each device is assigned a $\boxed{\text{device class name}}$ and a $\boxed{\text{logical device number}}$. For example, the card reader may be in device class CARD and be assigned logical device number 7.

When you tell MPE that you want to read data from a device having the device class name CARD, the data which is read from this device is considered a $\boxed{\text{devicefile}}$. Each card that is read from the card reader is a record. If a program sends information to a line printer, this data is also a devicefile.

Line Printer

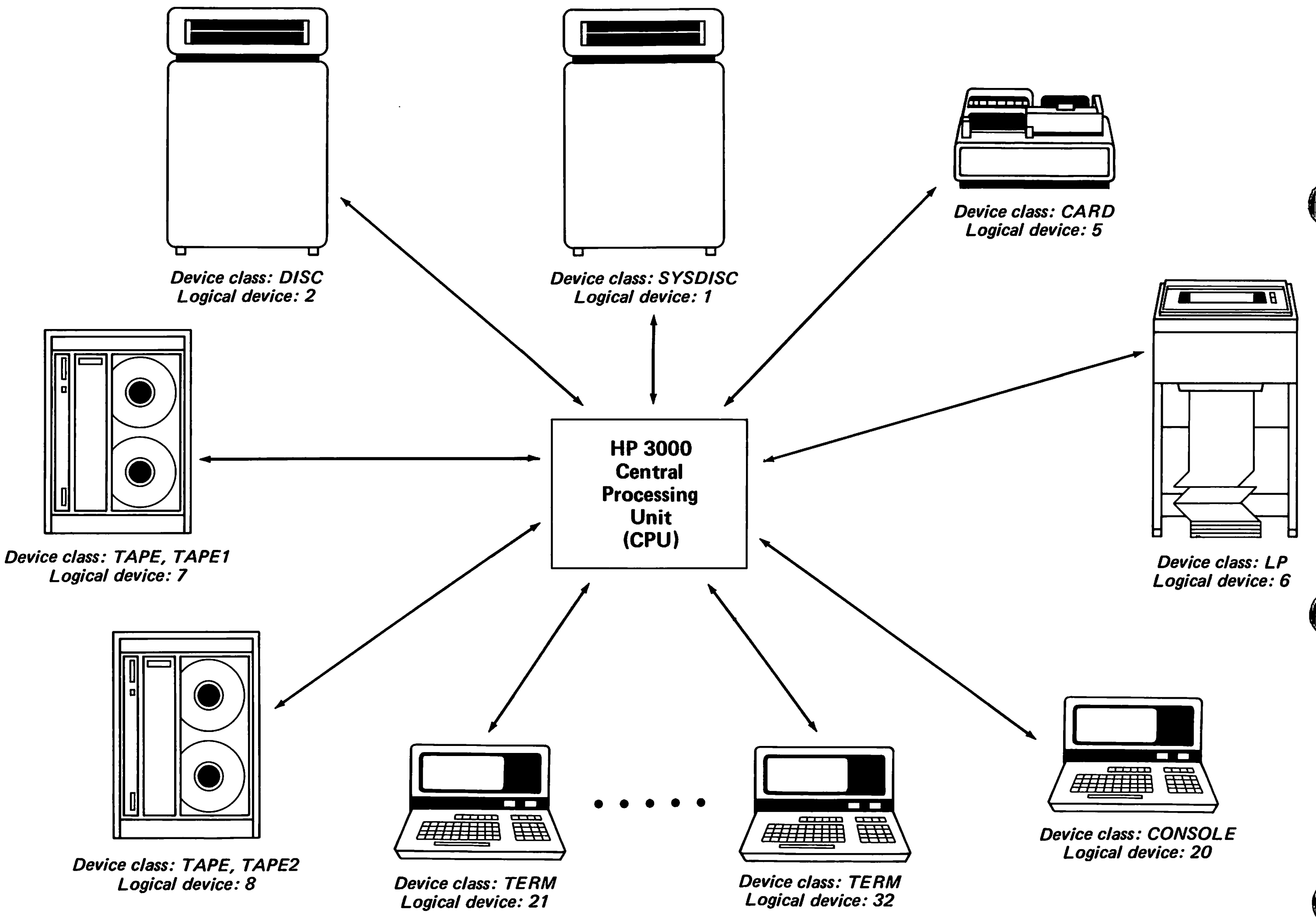


Although a disc drive is also a device, disc files are not called devicefiles in the MPE operating system. In fact, devicefiles are sometimes referred to as non-disc files.

In addition to one or more device class names and a logical device number, the system supervisor specifies characteristics for each device in the system including the following:

- device type and subtype (two system defined numbers that identify exactly what kind of device it is; for example, a hardwired terminal versus a terminal connected through a modem).
- record width
- whether the device can accept jobs, sessions, or data that is not part of a job or session (using the :DATA command that is discussed later in this guide).

For the purposes of this guide, let's assume our system is configured like this:



In most situations, you can refer to these devices by either their device class names or logical device numbers but the device class name is least likely to change.

how files are accessed

All MPE file operations are performed through a set of system-supplied procedures (intrinsics). These intrinsics are described in the *MPE Intrinsics Reference Manual.* If you are programming in COBOL, BASIC, FORTRAN, RPG, or APL, you don't have to call these intrinsics because the compiler (or interpreter) calls them for you when you use input-output statements of the language. Likewise, when you use utility programs provided by the system, you do not need to be concerned about these intrinsics. However, it is important to be aware of the way files are opened and closed in order to understand how to properly use the :FILE command.

The characteristics of a *new* file are determined at the time the file is opened and are defined by the parameters of the FOPEN intrinsic. They may be modified by a :FILE command entered during the session or job that refers to the same formal file designator as the FOPEN intrinsic. The physical characteristics of an existing (old) disc file cannot be modified; for example, it's record size or format. However, the :FILE command can be used to modify other characteristics such as file access restrictions. Illustrations of the :FILE command are provided in the examples which follow.

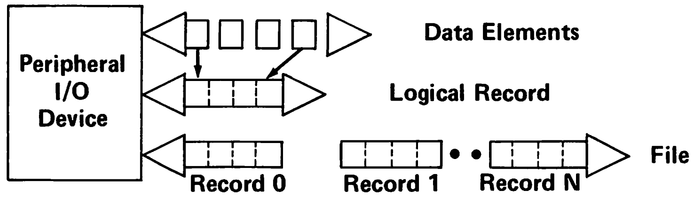
The decision to keep or discard a file created by the FOPEN intrinsic or an existing disc file is made when the file is closed. At that time the file may be saved as a permanent file or a session/job temporary file or it may be deleted. The file's disposition is specified as a parameter of the FCLOSE intrinsic or a :FILE command parameter which may have the value:

● SAVE	make the file a permanent disc file
● TEMP	make the file a temporary disc file to be deleted at the end of the current session or job
● DEL	delete the file.

A COBOL or BASIC program that performs operations on KSAM files must do so through calls to KSAM procedures which in turn call the MPE intrinsics.

file characteristics

A file consists of logically related records which in turn consist of logically related data elements.



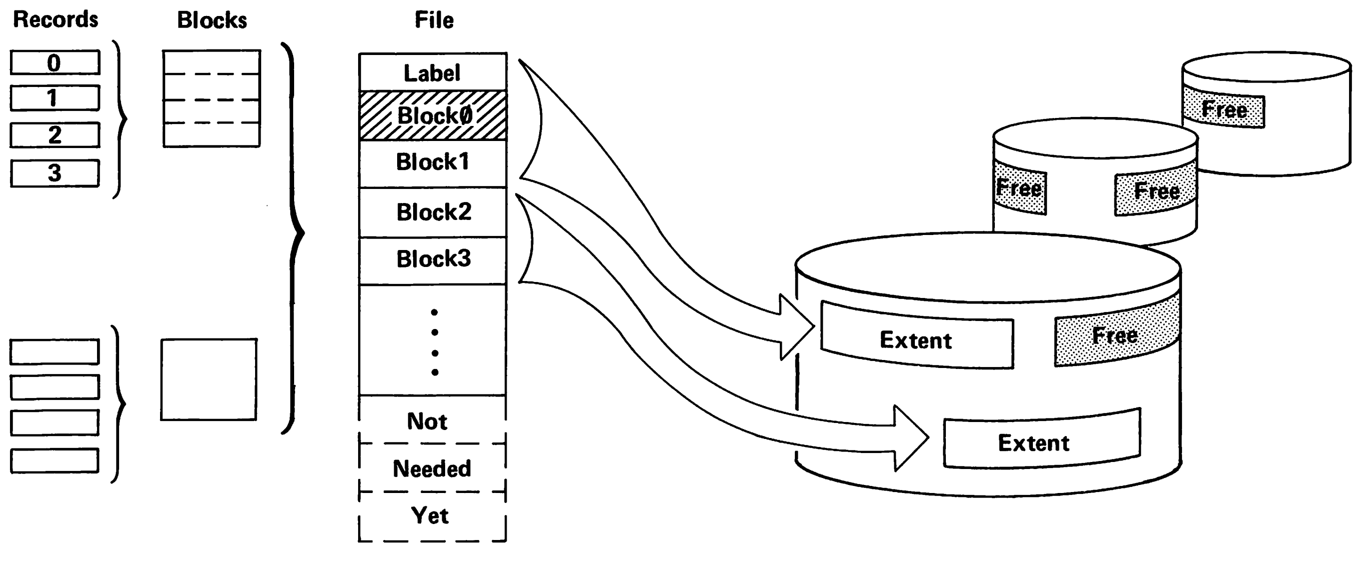
Logical records in disc and tape files may be grouped in blocks which are sometimes called physical records. Data is transferred to and from a device, one complete block at a time. Blocks for all other devices consist of 1 logical record per block.

Disc files may be divided into as many as 32 extents (16 for Pre-Series II systems). Extents allow you to use the disc more efficiently by allocating disc space as it is needed and making it easier to find enough consecutive disc space to accommodate a large file. For example, a file consisting of 10,000 records may be divided into 10 extents, each consisting of 1000 records. You may request that only 1 extent of 1000 records be allocated at first.

You need not be concerned with the actual physical location of the records or extents unless you are trying to optimize an application's disc access. The space for a file can be allocated one extent at a time, as it is required.

Logical records may be

- fixed length
- variable length, or
- undefined length (not used for disc normally).



Examples in this guide deal only with fixed-length records. See the *MPE Commands Reference Manual* or the *MPE Intrinsics Reference Manual* for a description of variable or undefined length records.

A file can contain ASCII (American Standard Code for Information Interchange) coded

- text
- source programs
- job control commands (with or without data)

or BINARY information such as

- compiled and prepared (executable) programs
- data.

You will see examples of these files as you read further in this guide.

The specification of a file as ASCII or BINARY does not perform any conversion of the data. ASCII coded data may be stored on a BINARY format file and vice-versa. However, to avoid confusion this is not usually done. If you tell the system that a file is ASCII format, it pads records preceding the end-of-file with blanks. If you tell the system that a file is BINARY format, it pads records with zeros.

EBCDIC or BCD coded data may be stored in either BINARY or ASCII files.

You may access disc files sequentially or directly (by logical record number).

FILE IDENTIFICATION

A file may be identified by a name. Disc files are also identified by a system file label and may optionally be identified by user labels and a file code.

A disc file name consists of at most 8 alphanumeric characters and must begin with an alphabetic character: for example, TESTFILE, INVENTRY, FILEZ, and MATLISTX. Devicefiles are sometimes referenced by special system-defined names that begin with a dollar sign ($). These files are called $\boxed{\text{system-defined files}}$ and are discussed in the next section.

File names may be qualified by the name of the group and account to which the file belongs as shown in the File Security section which follows.

The system file label is maintained automatically by the system and contains descriptive information about the file. If you create a disc file with user labels you are responsible for the content of these labels.

A file code is a number in the range of 0 to 1023 that identifies a group or class of files. The file code is saved in the system file label. File codes larger than 1023 are assigned special meanings by the system. (See the :BUILD command in Section II of the *MPE Commands Reference Manual* for a list of these codes and their meanings.) The use of file codes in your own files is completely optional. If you do not specify one, it will be zero by default. They can be useful for an application; for example, to identify the specific type of information that a file contains. You can retrieve the value of a file code programmatically by calling the FGETINFO intrinsic or using the :LISTF command as shown later in this guide.

using devicefiles

Your interaction with devicefiles will usually consist of entering commands like these:

:FILE PRINTER; DEV=LP	Tell MPE you are referencing a line printer by the formal and actual file designator PRINTER.
:FILE XXXOUT; DEV=TAPE; REC=200,3,F,BINARY	Tell MPE you are equating the formal file designator XXXOUT to a magnetic tape device with fixed length records 200 words long, containing binary data and written with three logical records per block.
:FILE XXXIN; DEV=CARD	Tell MPE you are referencing a card reader by the formal and actual file designator XXXIN.

You may also want to reference one of the system-defined files: $STDIN, $STDINX, $STDLIST, $OLDPASS, $NEWPASS, or $NULL.

:FILE XXXOUT=$STDLIST	Tell MPE to display the output file referenced in your program as XXXOUT on the $STDLIST device.
:FILE FTN06=$NULL	Tell MPE to discard all data written to the file named FTN06 by writing it to the system "ghost file," $NULL.

System defined files are actually just reserved words that refer to a specific type of system file.

• $STDIN	refers to the device that you used to initiate your current session or job. The device is normally a terminal for a session and a card reader for a job. ($STDINX is a special $STDIN file that can accept most MPE commands as data. See the *MPE Commands Reference Manual* for a complete description.)
• $STDLIST	is the device designated as the session or job output device, the device MPE uses to respond to your commands. This device is normally a terminal for sessions and line printer for jobs.
• $OLDPASS $NEWPASS	reference special temporary files. You will find an example of their use later in this guide.
• $NULL	is a file designator that is used to tell MPE to read from or write to a non-existent file as though the input-output operation were successful. This file is usually used to discard output.

Here are some things to remember about devicefiles:

- A devicefile as identified by a specific file designator exists only temporarily, from the time you open the file until you close it.
- With the exception of magnetic tapes, devicefile blocks contain only one logical record.
- Access to a device file is always sequential.
- You must specify the device class when using a :FILE command to define a devicefile, since the system default device class is DISC. (If an application program referencing the file specifies the device class when the file is opened, it can be omitted from the :FILE command.)

- You must protect devicefile data since the system does not provide security for data contained in cards, on magnetic tape, or line printer listings.
- Non-sharable devices such as card readers and line printers may be $\boxed{\text{spooled}}$. Spooling permits more than one job or session to gain access to a device without waiting by copying the data coming from or going to a spooled device onto a disc until the device is free, and then completing the input-output operation requested.

Device class names are determined by the system manager and vary from installation to installation. Thus, the line printer may be called LP on one system, PRINTER on another. Before you attempt to use these names, find out what device class names have been defined for your installation.

file security

Devicefile data such as that stored on magnetic tape or cards is controlled by those who generate it. The data can be locked in cabinets and physically protected from unauthorized access. Disc file data, on the other hand, resides on one or more discs shared by all the system's users. In order to protect the files you create, MPE manages the system resources and controls access to the system through accounts, groups, and user names. Each file belongs to a $\boxed{\text{group}}$ within an $\boxed{\text{account}}$ and to the $\boxed{\text{user}}$, or member of the account, who creates the file. To further protect a file, the creator may assign a $\boxed{\text{lockword}}$ to it.

The file system determines whether or not to grant access to a file when a request is made. You must meet the security requirements at three levels, the

- account level
- group level
- file level

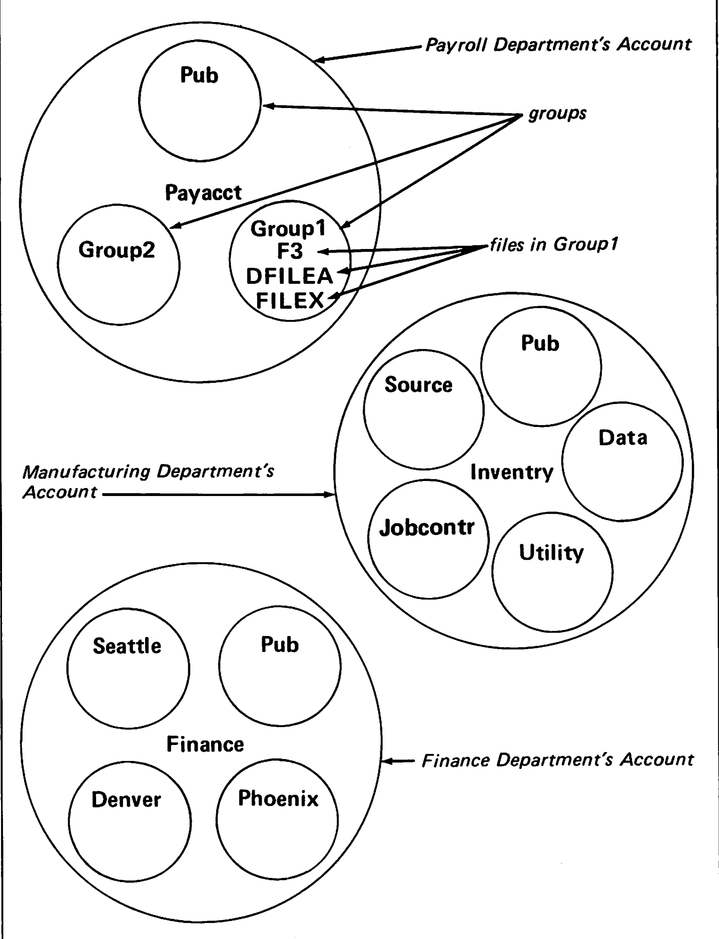
The creator of a file has additional capabilities such as altering the file's lockword.

The HP 3000 account structure helps you protect the files you create and provides a means of controlling and monitoring system usage. You can define a very simple structure initially; for example, your system may initially consist of the SYS account for general use – you can add to its complexity as you become familiar with the various options and want to use them.

Each account can use its groups for a different purpose. In the example:

- The FINANCE account keeps files related to different divisions of the company in separate groups.
- The INVENTRY account separates files by type. All data files are kept in a group named DATA; all program source files in a group named SOURCE; all job control files in the JOBCONTR group; and all utility routines in the UTILITY group.
- The PAYACCT separates files used by one project group from those used by another project group.

ACCOUNTS AND GROUPS

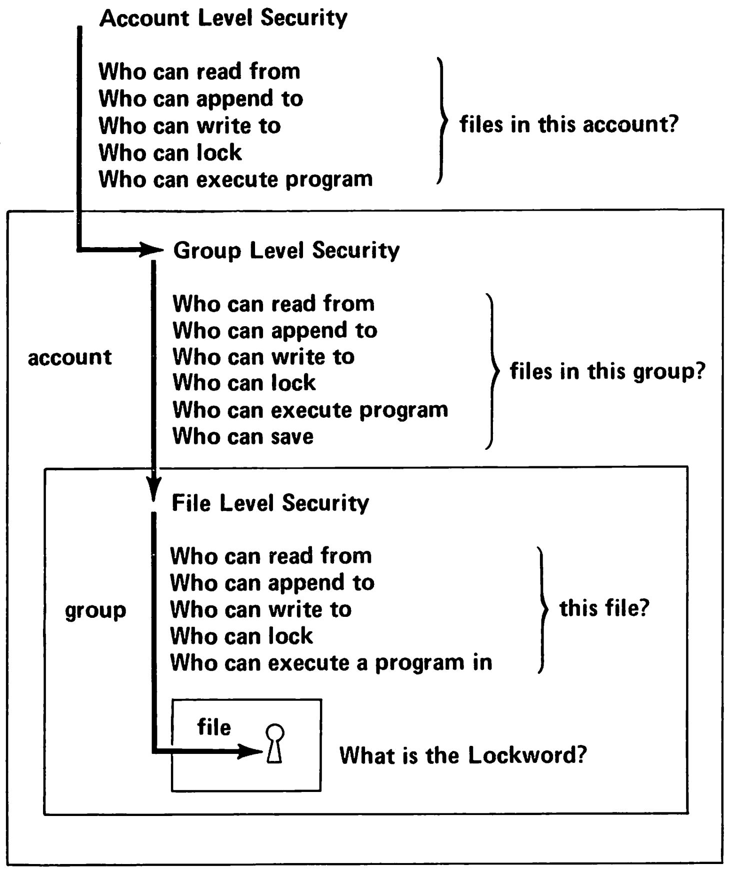


The accounting structure illustrated here consists of three accounts:

- PAYACCT
- FINANCE
- INVENTRY

A SYS account must exist as part of the system also.

The diagram below illustrates the protective barriers that you may establish for a file. These levels form a hierarchy or series of filters. Before you can access a file, you must first pass the account level, then the group level, and finally the file level of security.



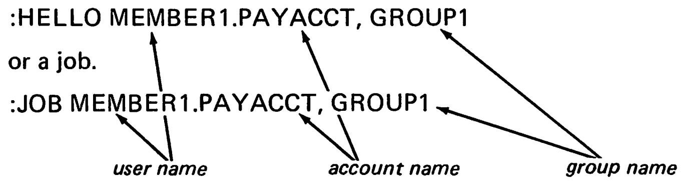
ANY	— any system user?
AC	— an account user?
AL	— an account librarian?
GU	— a group user?
GL	— a group librarian?
CR	— the file's creator?

You establish your log-on account and log-on group when you initiate a session

```
:HELLO MEMBER1.PAYACCT, GROUP1
```

or a job.

```
:JOB MEMBER1.PAYACCT, GROUP1
```



When the account manager defines your user name, he may optionally assign a home group to you. If you do not specify a group name when using the :HELLO or :JOB commands, your log-on group is the same as your home group.

MEMBER1's home group is GROUP1.

```
:HELLO MEMBER1.PAYACCT
        .
        .
        .
:FILE INFILE = FILEX
```

If you do not have a home group, you must include a group name in the :HELLO or :JOB command. If you have a home group and specify some other group when you log on, you are considered to be a group user (GU) of both your home group and your log-on group. However, you must indicate when you want to use a file that is not in the log-on group by qualifying the file name as shown below.

If you do not initiate your job or session with the names of the account and group to which a file belongs, you may be able to gain access to the file by using a fully-qualified file name when you reference it. For example,

```
:FILE INFILE = FILEX.GROUP1.PAYACCT
```

File name qualified by group and account names.

If you are using the account but not the group to which the file belongs, you only need to qualify the file name with the group name.

MEMBER1 whose home group is GROUP1 logs on to GROUP2.

```
:HELLO MEMBER1.PAYACCT, GROUP2
                  .
                  .
                  .
:FILE INFILE = FILEX.GROUP1
```

MEMBER1 qualifies the file name with the name of his home group.

Usually you will be logged on to or running jobs in the file's account and group and can simply use the file name (and its lockword if one is assigned).

```
:FILE INFILE = FILEX
```

It is also possible to gain access to a file by asking the file's creator to temporarily release (turn off) the file's security provisions using the :RELEASE command. The creator then must use the :SECURE command to restore the file's security provisions when you are finished using it.

The security restrictions for an account are defined by a system manager when the account is opened and may be altered by a system manager. The security restrictions for a group are defined, and may be altered, by an account manager. When you create a file, its security is defined automatically by a set of default restrictions. You may alter these restrictions by using the :ALTSEC command.

If no security restrictions are specified when the account and group are defined—by default, the security requirements are the following:

- Account Level — Only account members can access the files in the account and they are granted all types of access.
- Group Level — Files in the PUB group may be read, and program files in this group may be executed, by any user who passes the account level of security. Only account librarians and group users are allowed to append to, write to, and lock files in the PUB group. Only account librarians and group users are allowed to save files in the PUB group.

  Files in any other group may be accessed only by group users, and they may access them in any way.
- File Level — Any user who passes the account and group levels of security can access the file in any way but they must provide the correct lockword if one is defined.

Here is a summary of these provisions:

	Account Level		**Group Level**		**File Level**
	SYS Account	Other Accounts	PUB Group	Other Groups	All Files
R	ANY	AC	ANY	GU	ANY
A	AC	AC	AL or GU	GU	ANY
W	AC	AC	AL or GU	GU	ANY
L	AC	AC	AL or GU	GU	ANY
X	ANY	AC	ANY	GU	ANY
S			AL or GU	GU	

R = read files
A = add data records to file but cannot delete or alter existing data
W = add, delete, or change data or delete files
L = lock files
X = run programs stored in files
S = save permanent files within a group

ANY = any user	AC = account user
AL = account librarian	GU = group user

CREATOR CAPABILITIES

The following capabilities are granted only to the file's creator:

- renaming the file
- changing the lockword of a file
- altering the file level security restrictions
- releasing or suspending the file's security restrictions
- securing these restrictions again.

The creator can alter the file level security restrictions in such a way that only he or she has certain types of access to the file.

For more information about defining accounts and groups, see the *System Manager/System Supervisor Reference Manual.*

KSAM files

In addition to regular disc files, you may want to use the file handling capabilities available through the KSAM/3000 (Keyed Sequential Access Method) software.* KSAM offers all the MPE file system capabilities and in addition makes it possible to create and maintain disc files whose records can be accessed by the value of key fields within the data records.

A KSAM file is organized into two distinct MPE files, a data file and a key file. The key file contains only key entries, the data file only data. Each record in the data file contains at least one item that is designated as a key. The value of each key is duplicated in the key file where all keys are ordered in ascending sequence. The key file maintains the logical order of the records even though they may not be stored in sorted order in the data file.

Although a KSAM file consists physically of two separate files, a data file and a key file, it is treated as one file for most purposes. For example, reading from a KSAM file in primary sequence is equivalent to reading sequentially from a non-KSAM file. Similarly, creating the data file portion of a KSAM file is equivalent to creating a non-KSAM file.

Most of the examples in this manual illustrate MPE file operations although there are a few examples of KSAM file manipulation. The major differences in manipulating MPE and KSAM files are:

- When you create, rename, save, or delete a KSAM file, you must use the utility program KSAMUTIL rather than the MPE commands.
- When you use the FCOPY utility program, you may use some special FCOPY parameters that relate only to KSAM files.

There are many examples of using KSAM files in the *KSAM/3000 Reference Manual* and the *RPG/3000 Compiler Reference and Application Manual.* Consult these manuals for detailed information about KSAM files.

*Available only on HP 3000 Series II Computer Systems.

as you read the examples...

Note that the methods illustrated are not the only ones for accomplishing the tasks. When you are familiar with the system you may find other ways that you prefer to accomplish some of the tasks.

You can determine syntax rules for the commands illustrated by following the spacing and punctuation shown in the example. If you want complete syntax rules, consult the appropriate reference manual for the commands or subsystems being illustrated. In general, spaces are allowed between parameters but not in the middle of parameter values such as files names or reserved words. If you are unsure of a syntax rule, it is usually safe to experiment if you are not operating in production mode.

To distinguish user responses from system prompts and messages in the examples, user input is printed in blue.

If you decide to try some of these examples and an error occurs, you may see something like this displayed on your terminal:

```
INVALID FILE REFERENCE (FSERR 54)
```

If a file error occurs before a file is successfully opened, you may see a message like this:

```
+-F-I-L-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y+
!  ERROR NUMBER: 52     RESIDUE: 0                !
!  BLOCK NUMBER: 0             NUMREC: 0          !
+-------------------------------------------------+
                       error
```

In this case, look up the error message number in Table 2-7 of the *Error Messages and Recovery Manual.* Usually the utility program you are running allows you to continue after a File Information Display is printed, but you must determine the cause of the error and correct it.

If an error occurs after a file is open, the File Information Display provides more information about the file:

```
                                 file name
+-F-I-L-E---I-N-F-O-R-M-A-T-I-O-N---D-I-S-P-L-A-Y+
! FILE NAME IS IN.PUB.TECHPUBS                    !
! FOPTIONS: NEW,A,*FORMAL*,F,N,FEQ                !
! AOPTIONS:   INPUT,SREC,NOLOCK,DEF,BUFFER        !
! DEVICE TYPE: 0       DEVICE SUBTYPE: 3          !
! LDEV: 17       DRT: 12         UNIT: 1          !
! RECORD SIZE: 256     BLOCK SIZE: 256    (BYTES) !
! EXTENT SIZE: 128     MAX EXTENTS: 8             !
! RECPTR: 0            RECLIMIT: 1023             !
! LOGCOUNT: 0              PHYSCOUNT: 0           !
! EOF AT: 0            LABEL ADDR: %02100060123   !
! FILE CODE: 0       ID IS MAC        ULABELS: 0  !
! PHYSICAL STATUS: 1111000000000001               !
! ERROR NUMBER: 40     RESIDUE: 0                 !
! BLOCK NUMBER: 0              NUMREC: 1          !
+-------------------------------------------------+
                               error
```

Until you gain more experience with the system, look up the error number in Table 2-7 of the *Error Messages and Recovery Manual.* Later, you may find the other information useful. It is described in Appendix A of the *Error Messages and Recovery Manual.*

Now, let's begin.

defining accounts, groups, and users

The system manager first creates an account. In this example, an account is set up in the standard way with no special capabilities or restrictions. Each account must have an account manager. A PUB group is automatically created.

```
:HELLO MANAGER.SYS
ACCT PASSWORD?
MISHTO
```

System manager logs on to the system. The system prompts for the password.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

The system prints log-on information.

```
:NEWACCT PAYACCT,MANAGER
:BYE
```

System manager defines a new account named PAYACCT with account manager named MANAGER and then logs off the system.

```
CPU=1.  CONNECT=1.  MON, JAN 9, 1978, 8:03 AM
```

The account manager may now define groups and users. In this example, the account manager defines a group and user with standard capabilities and restrictions with the exception that a home group is assigned to the user.

```
:HELLO MANAGER.PAYACCT
```

Account manager logs on to the system.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:NEWGROUP GROUP1
:NEWUSER MEMBER1; HOME=GROUP1
:HELLO MEMBER1.PAYACCT
```

Account manager defines new group named GROUP1. and a new user named MEMBER1 whose home group is GROUP1. The new user may now log on with a username and account name.

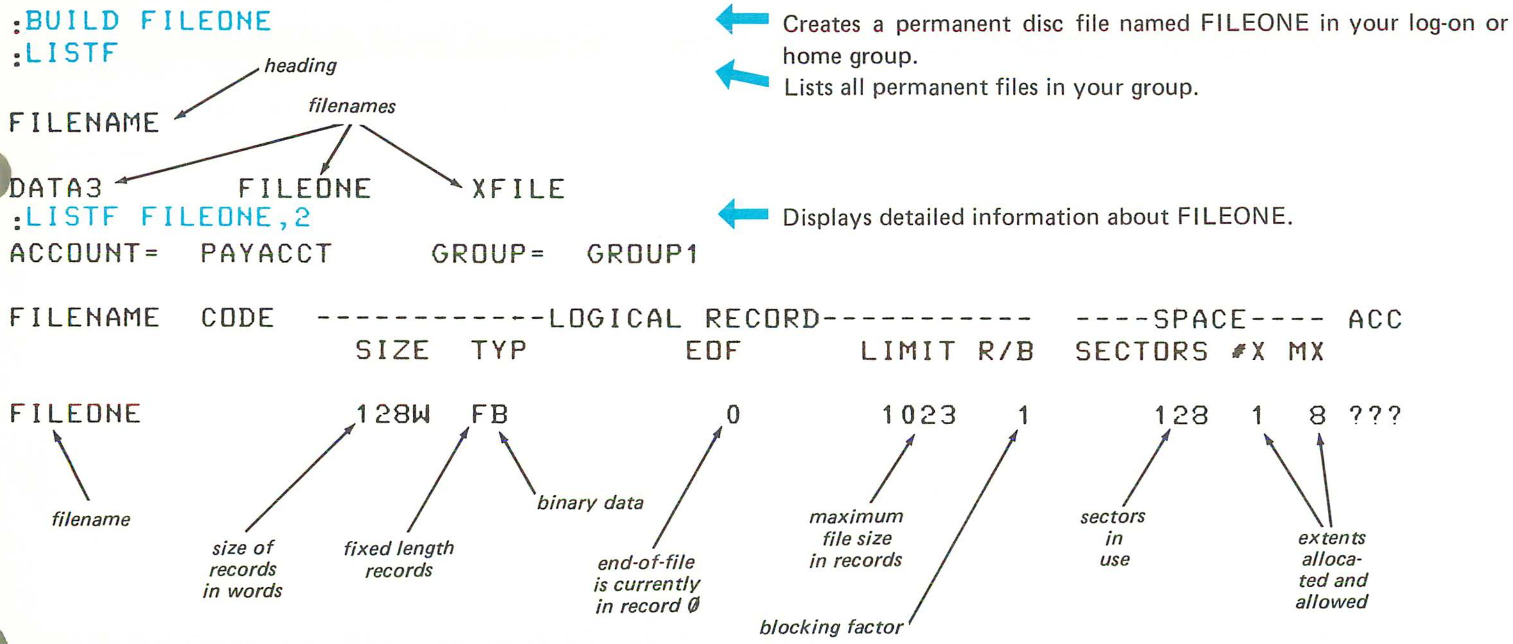
ADDITIONAL INFORMATION:

System Manager/System Supervisor Reference Manual (Section IV and V)
MPE Commands Reference Manual (Section III)

creating disc files

creating a disc file (using :BUILD)

You can use the :BUILD command to create a disc file immediately. If you want to verify that you created the file, you can use the :LISTF command.



```
:BUILD FILEONE
:LISTF

FILENAME

DATA3              FILEONE          XFILE
:LISTF FILEONE,2
ACCOUNT=   PAYACCT          GROUP=   GROUP1

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

FILEONE           128W   FB             0       1023   1      128   1   8 ???
```

:BUILD FILEONE — Creates a permanent disc file named FILEONE in your log-on or home group.

:LISTF — Lists all permanent files in your group.

:LISTF FILEONE,2 — Displays detailed information about FILEONE.

A filename may be from 1 to 8 alphanumeric characters. The first character must be alphabetic.

Since all optional parameters are omitted in the :BUILD command shown in the example, the file assumes the default characteristics:

- 128 word or 256 character (byte) logical records
- fixed length records
- binary format
- maximum file size of 1023 logical records
- blocking factor of 1 (1 logical record is transferred to or from the disc at a time)
- 8 extents (the file is divided into 8 sections or extents, in this case, each containing 128 contiguous sectors or physical records)
- 1 extent is allocated initially
- the file code is 0
- no carriage control information is to be provided when the file is used.

The following :BUILD command creates a file with these same characteristics: :BUILD FILEONE; REC=128,1,F,BINARY; NOCCTL; DEV=DISC; CODE=0; DISC=1023,8,1

Some users prefer to specify all the parameters of a command, even though they are using the default values, in order to document the characteristics of the file.

This is particularly useful when a command is part of a set of job control commands executed in batch mode on a production basis.

The TEMP; parameter of the :BUILD command has been omitted in both examples; therefore FILEONE is a permanent file. It is saved immediately and retained after the session or job terminates.

The :BUILD command

- allocates space for the file on the disc
- sets an end-of-file pointer to the beginning of the file.

If you try to read the file, you will be notified that it is empty. It is ready to have data entered into it.

The :LISTF command displays information about files.

- If no parameters are specified, it lists the names of all permanent files in your log-on group.
- If the file name, a comma, and a 2 are specified, it lists information about the named file. Less information is displayed if you specify 1, instead of 2.

You cannot create a file in an account unless it is the account in which you are running your session or job.

You can create a file in a group that is not your log-on group, if the security provisions of that group allow users who are not logged on to the group to save files in the group, or if it is your home group.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

creating a disc file with specific characteristics

Usually you will want to specify some special characteristics for the file you are creating – for example, a particular record size or format.

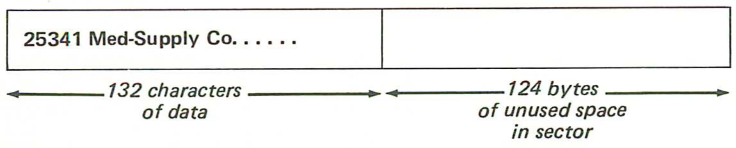
Minus sign indicates quantity or length is in characters (bytes not words)

```
:BUILD FILETWO; DISC=200,2; REC=-132,,,ASCII
:LISTF FILETWO,1
ACCOUNT=   PAYACCT      GROUP=   GROUP1

FILENAME   CODE   ------------LOGICAL RECORD--------
                      SIZE   TYP         EOF        LIMIT

FILETWO               132B   FA            0          200
```

parameters omitted

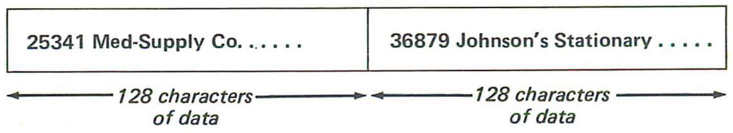
Create a disc file named FILETWO with 200 records, each 132 characters long, ASCII format. The file is divided into 2 extents; by default, 1 extent is allocated.

After you use FILETWO to store ASCII information, a physical record might look like this:


25341 Med-Supply Co. . . . . .
132 characters of data
124 bytes of unused space in sector

132 + 124 = 256 characters (bytes) in disc sector

Since the parameters are omitted, the blocking factor is 1 and the records are fixed length by default. Thus, 1 fixed length record is to be moved to the buffer at a time and each disc sector contains only 1 logical record. In order to conserve disc space, it would be much better to design the file with 128 character records and a blocking factor of 2 (or a multiple of 2). Then the logical records and disc sector would look like this:


25341 Med-Supply Co. . . . . .
36879 Johnson's Stationary . . . . .
128 characters of data
128 characters of data

128 + 128 characters (bytes) in disc sector

And the :BUILD command would look like this:

:BUILD FILETWO; DISC=200,2; REC=-128,2,,ASCII

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Sections VI and VII)

creating a disc file while a program is executing

You can create a file at the time your program opens the file by using the :FILE command with the NEW parameter.

```
:FILE OUT=FILEONE,NEW; REC=40,16,,ASCII; SAVE
:RUN APPLICN
```

When the APPLICN program opens a file it calls OUT, create a new permanent disc file named FILEONE.

In this example:

- the logical record length is 40 words or 80 characters
- the blocking factor is 16
- the format of the data is ASCII
- the SAVE parameter makes the file a permanent one.

Although it is possible to determine the characteristics of a file created with the :BUILD command by examining the parameter values supplied with the command and knowing the default values for the optional parameters, it is not as simple to do this with the :FILE command. The type of file that is created depends not only on the :FILE command default values but also on the programming language used. If a :FILE parameter is omitted, and the program or compiler does not specify the parameter value, the MPE default values are used. For example, if APPLICN is a COBOL program and the *system-file-name* in the ASSIGN clause of the SELECT statement does not specify a *filesize,* the COBOL compiler specifies a file size of 10000 records. If you omit the DISC= *numrec* parameter from the :FILE command, the file is created with 10000 records rather than 1023 (the system default).

If APPLICN is an RPG program, the *filesize* is 1023 (the system default) since RPG does not allow you to specify the file size within the program.

If you only specify :FILE OUT = FILEONE, NEW, the system defaults for a complete :FILE command are:

```
:FILE OUT = FILEONE, NEW; REC = 128, 1, F, BINARY;
NOCCTL; DEL; DEV = DISC; CODE = 0; DISC = 1023, 8, 1;
ACC = IN; BUF = 2; SHR; NOMULTI; NOMR; WAIT
```

Note that if you do not specify whether a file is NEW, OLD, or OLDTEMP, FORTRAN opens the file as a NEW file. If you want to reference an existing disc file you must specify OLD for a permanent file and OLDTEMP for a temporary file.

In the previous example, the SAVE parameter is used to override the DEL default for files whose domain is NEW.

If file characteristics are specified in the program that opens the file, the system defaults are not used, and in some cases, the programming language has its own file characteristic defaults. This table indicates how the characteristics are determined according to the language in which the program is written:

Characteristic	COBOL	RPG	FORTRAN	BASIC **
REC=*recsize*,	File Description (FD) statement RECORD CONTAINS n	File Description (FD) specification cols. 24-27 (80 bytes default)	128 if fixed-length	128
blockfactor,	BLOCK CONTAINS n	FD cols. 20-23 and 24-27 (1 default)	1	1
F or U or V,	RECORDING MODE IS n	FD col. 19 (F default for disc)	V for sequential F for direct access	F
BINARY or ASCII;	SELECT statement *system-file-name* (ASCII default)	ASCII	BINARY except FTN05, FTN06	BINARY
COTL or NOCCTL;	SELECT statement *system-file-name* (NOCCTL default)	Control Record specification carriage control then CCTL	NOCCTL except FTN06	NOCCTL
DEL or SAVE or TEMP;	DEL	TEMP	DEL	DEL
DEV=*device*;	SELECT statement *system-file-name* (DISC default)	FD cols. 40-46 (DISC default)	DISC except FTN05, FTN06	DISC
CODE=*filecode*;	0	0	0	0
DISC=*numrec*,	SELECT statement *system-file-name* (10,000 default)	1023	1023	1023
numextents,	8	FD cols. 68-69	8	8
initalloc;	1	1	1	1
ACC=IN or OUT or UPDATE or OUTKEEP or APPEND or INOUT;	OPEN statement INPUT = IN OUTPUT = OUT I-O = UPDATE	Depends on file type FD col. 15 and col. 66	INOUT	ASSIGN statement *restriction* parameter (INOUT default)
NOBUF or BUF=*numbuffers*;	SELECT statement RESERVE clause (2 default)	BUF=2	BUF=2	BUF=2
EXC or EAR or SHR	If INPUT then SHR else EXC	If locking, SHR else EXC	EXC	ASSIGN statement *restriction* parameter (EAR default)

** If you use the SYSTEM *numeric variable,* "FILE . . . ." statement, the parameters you specify will determine these file characteristics.

The MULTI/NOMULTI, MR/NOMR, and NOWAIT/WAIT parameters are used in advanced applications and are explained in the *MPE Command Reference Manual.*

What are the advantages and differences between the :BUILD and :FILE commands?

:BUILD	**:FILE *name*, NEW**
Allocates space for a new disc file, places the file name in a directory, initializes the file to blanks (ASCII) or zeros (BINARY), writes an end-of-file mark at the beginning of the file and sets the file pointer at the beginning of the first record. The file is a permanent file by default but may be session/job temporary if you specify TEMP.	Performs the same operations as the :BUILD command but does so when the program opens the file. By default (DEL) the file will not be saved but may be saved as a permanent or session/job temporary file if you specify SAVE or TEMP.

The :FILE command's primary purpose is to change a file's characteristics, overriding the characteristics defined in a program when the file is opened and closed.

You can equate the formal file designator used in the program to an actual file designator, change the file code, or change most of the characteristics shown in the preceding table of languages.

You cannot change characteristics recorded in the label of an existing file. These characteristics include the file size, record size, and block size.

The :FILE command is also used to:

- define devicefiles
  :FILE PRINT; DEV=LP
- provide special run-time file control; for example, input/output access and sharing the file
  :FILE XXX; ACC=INOUT; SHR
- equate a file to a system-defined file
  :FILE XXX=$NULL
- back reference another :FILE command.
  :FILE XXX=*YYY

(Examples of this technique appear later in this guide.)

ADDITIONAL INFORMATION:

BASIC Interpreter Reference Manual
MPE Commands Reference Manual (Section VI)
COBOL/3000 Reference Manual
FORTRAN Reference Manual
RPG/3000 Compiler Reference and Application Manual

creating a disc file by copying another file

FCOPY, the file copier utility program, allows you to create a new file and copy information to it from an existing file.

```
:RUN FCOPY.PUB.SYS
```

Run FCOPY program.
(Program is in PUB group of SYS account.)

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=EMPLOYEE; TO=EMPFILE; NEW
```

Copy contents of EMPLOYEE file to new file named EMPFILE.

```
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS
```

FCOPY indicates that end-of-file has been encountered and tells how many records have been copied.

```
>EXIT
```

Terminate FCOPY execution.

```
 END OF PROGRAM
:LISTF EMPLOYEE,2
```

List information about EMPLOYEE file

```
ACCOUNT=  PAYACCT      GROUP=  GROUP1

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

EMPLOYEE        108B  FA          20         20  16        2  1  1 ???

:LISTF EMPFILE,2
```

List information about new file.

```
ACCOUNT=  PAYACCT      GROUP=  GROUP1

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

EMPFILE         108B  FA          20         20  16       21  1  1 ???
```

As you can see from the :LISTF output, the new file has the same characteristics as the original file. FCOPY gives the TO-file the characteristics of the FROM-file unless you specify otherwise by using a :FILE command as shown in some of the examples that follow.

To make an FCOPY command easier to read, you may include blanks between the elements.

FCOPY, like MPE, references the first record as record number 0. Therefore, record 19 is the 20th record copied.

ADDITIONAL INFORMATION:

FCOPY Reference Manual

creating a file with the editor

You can create files of ASCII data with the Editor, EDIT/3000. This technique is often used to create source programs and job control command streams.

```
:EDITOR
```

Initiate execution of Editor by entering the MPE :EDITOR command.

```
HP32201A.5.01 EDIT/3000  TUE, JAN 18, 1977,   8:27 AM
(C) HEWLETT-PACKARD CO. 1976

/ADD
```

Request to add lines to the work file Editor automatically creates.

```
    1       TEXT OF FIRST RECORD IN FILE
    2       TEXT OF SECOND RECORD IN FILE
    3                 .
                      .
                      .
```

Enter data when Editor prompts for a line (or record)

To stop adding lines, press control key and Y key simultaneously.

```
          ...
/KEEP EDITFILE,UNN
/EXIT
```

Save work file contents in file named EDITFILE omitting line numbers (unnumbered). Terminate Editor.

```
IF IT IS OK TO CLEAR RESPOND "YES"
CLEAR? YES
```

If the work file can be cleared, respond with yes.

```
 END OF SUBSYSTEM
```

The file has the following characteristics:

- 72 character records (a default value which you can modify)
- fixed length records (by default. You can request variable length records.)
- ASCII format
- the file size is equal to the number of records you create
- the blocking factor, number of extents, and number of extents allocated vary depending on the file and record size

To create a file with a different record size, use the Editor command SET LENGTH=n.

```
:EDITOR

HP32201A.5.01 EDIT/3000  WED, JAN 19, 1977, 11:18 AM
(C) HEWLETT-PACKARD CO. 1976
/SET LENGTH=80,RIGHT=80
/ADD
    1       !JOB MEMBER1.PAYACCT
    2       !COMMENT ********************
    3       !COMMENT     ACCOUNTS RECEIVABLE UPDATE
```

← Define a line (record) length of 80 characters, and right margin of 80 characters. Add lines to the work file.

You can add records to the file later by using the Editor TEXT and ADD commands.

```
:EDITOR

HP32201A.5.01 EDIT/3000  WED, JAN 19, 1977, 11:25 AM
(C) HEWLETT-PACKARD CO. 1976
/TEXT EDITFILE,UNN
/LIST LAST
   46      THIS IS THE LAST RECORD IN THE FILE
/ADD
   47      NEW DATA ADDED TO FILE
   48
   49
   50      ...
/K EDITFILE,UNN
EDITFILE ALREADY EXISTS - RESPOND YES TO PURGE OLD AND THEN KEEP
PURGE OLD?Y
/E
IF IT IS OK TO CLEAR RESPOND "YES"
```

← Move contents of EDITFILE into the work file.

← List the last line in the file.

← Add lines following the last line.

← Enter Control Y.

← Save the work file in EDITFILE.

← A new EDITFILE is created and the previous one is deleted. Terminate Editor.

The Editor commands may be abbreviated, for example, K for KEEP and E for EXIT or END.

ADDITIONAL INFORMATION:

EDIT/3000 Reference Manual

Using the HP 3000 (Section I)

creating a temporary disc file

When you create a file with the :BUILD command, it is a permanent file unless you include the TEMP parameter.

```
:BUILD FILETEMP; TEMP
```

Create a temporary file named FILETEMP.

```
:LISTF FILETEMP,2
```

Try to list information about the file. Since the file is not a permanent file it is not in the system file directory.

```
NONEXISTENT TEMPORARY FILE (FSERR 53)
:BUILD FILETEMP; TEMP
```

However, if you try to create another temporary file with the same name you cannot since the filename is in the session/job temporary directory.

```
DUPLICATE TEMPORARY FILE NAME (FSERR 101)
 HELLO MEMBER1.PAYACCT
```

Initiate a new session. (Current session is automatically terminated when you enter a new :HELLO command.)

```
CPU=2.  CONNECT=11. MON, JAN 9, 1978, 2:20 PM

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

Now you can create a file with the same name since all temporary files are deleted when a session (or job) terminates.

```
:BUILD FILETEMP; TEMP
:PURGE FILETEMP, TEMP
```

If you want to delete (purge) the file before the session terminates you use the :PURGE command with the TEMP parameter.

You can also create a temporary file at the same time the program opens the file by using the :FILE command, omitting the SAVE parameter and using the TEMP parameter.

```
:FILE OUT=FILETEMP,NEW;TEMP
:PURGE FILETEMP, TEMP
NONEXISTENT TEMPORARY FILE (FSERR 53)
:RUN APPLICN
            .
            .
```

Equate OUT file referenced in program to temporary file named FILETEMP. If you try to delete the file the system tells you it cannot find a temporary file with that name. When you use the :FILE command to create a file, you must access that file with a program or it will not be created.

```
:PURGE FILETEMP, TEMP
:
```

After the program executes, the file exists. (You would not usually bother to delete a temporary file as shown in this example.)

However, if you change your mind and want to make a session/job temporary file a permanent one, you can use the :SAVE command.

For example:

```
  :SAVE FILETEMP
```

If you do not specify TEMP or SAVE with the :FILE command, the file is deleted when the APPLICN program closes it since the default for this parameter is DEL (delete the file).

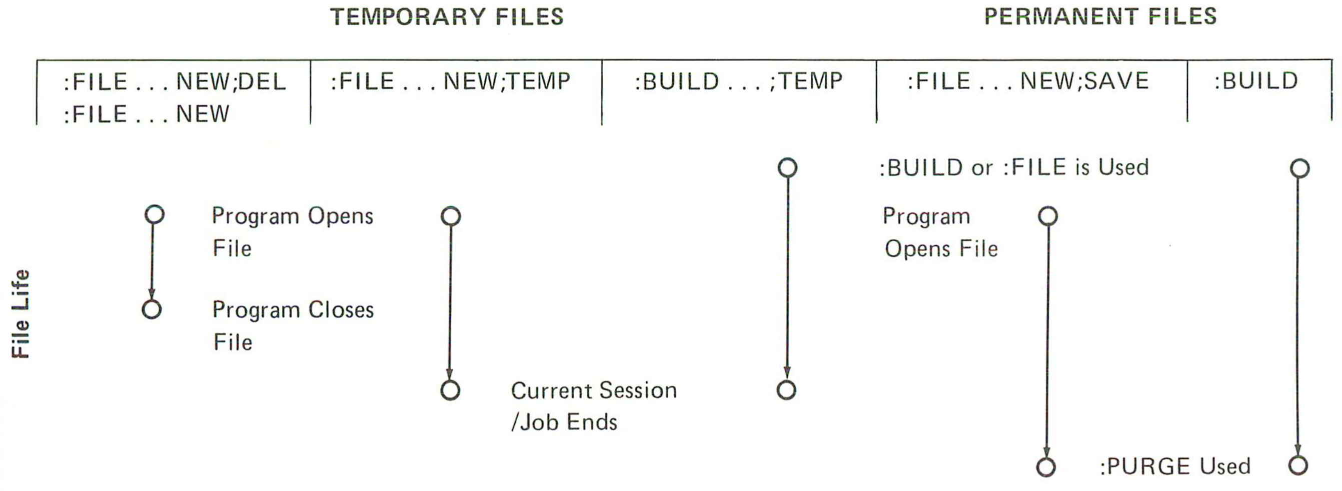
```
:FILE OUT=FILETEMP,NEW
:RUN APPLICN
                  .
                  .
                  .
:PURGE FILETEMP, TEMP
NONEXISTENT TEMPORARY FILE (FSERR 53)
```

Equate OUT file to a new file named FILETEMP and execute APPLICN program.

When program closes the file, it is deleted – therefore, it does not exist when the program terminates.

Job/session temporary files are catalogued in the Job Temporary File Directory. The files in this directory exist in the job/session file domain. Permanent files are catalogued in the System File Directory and exist in the system file domain. You may use the same name for more than one file if the files are not in the same domain.

You may also use this name for a file that exists only between the time a program opens the file and closes it. If the system is searching for an existing file, it checks the Job Temporary File Directory before the System File Directory.



You can list the names of temporary files by using the utility program, LISTEQ2* which is in the PUB group of the SYS account.

```
:BUILD XTEMP; TEMP                          ⟵ Create a temporary file named XTEMP.
:FILE CUMACR=FILEONE; ACC=APPEND            ⟵ Enter some :FILE commands.
:FILE PRINTER; DEV=LP
:RUN LISTEQ2.PUB.SYS                        ⟵ Initiate LISTEQ2 execution.

LISTEQ2  (C) COPYRIGHT HEWLETT-PACKARD COMPANY 1976.

***TEMP FILES                               ⟵ Names of temporary files are listed.

XTEMP.GROUP1.PAYACCT

***FILE EQUATIONS

:FILE CUMACR=FILEONE;ACC=APPEND             ⟵ LISTEQ2 also lists the :FILE commands in effect during the cur-
:FILE PRINTER;DEV=LP                          rent session or job.
```

If you want to clear the file characteristics assigned to a *formal* file designator, use the :RESET command.

```
:RESET CUMACR ⟵———————————— Clears file characteristics assigned by previous :FILE command.
```

You can reset or cancel all :FILE commands previously entered in the session by entering

```
:RESET @
:
```

*LISTEQ on pre-Series II systems.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)
MPE System Utilities Reference Manual (Section V)

changing file names and deleting files

deleting a disc file

If you no longer want to keep a disc file, you can delete it from your group and account by using the :PURGE command.

```
:LISTF                                                   ← List files in log-on group.

FILENAME

CUSTOMER    DATA3      EMPFILE     EMPLOYEE    FILEONE     FILETWO
FILEXXX     XFILE
:PURGE DATA3                                             ← Delete DATA3 file from group.
:LISTF                                                     List files again.

FILENAME

CUSTOMER    EMPFILE    EMPLOYEE    FILEONE     FILETWO     FILEXXX
XFILE
```

The permanent file named DATA3 is deleted from the log-on group. If you want to delete a temporary file, you must append a comma and TEMP to the file name, for example, :PURGE FILEX, TEMP. If the file has a lockword you must supply it following the file name and a slash, :PURGE FILEX/PASS, TEMP. In session mode, MPE prompts for the lockword if it is not supplied.

You can delete a file from any group or account to which you have write (W) file access capability.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

changing the name of a disc file

If you are the creator of a file, you can change its name. You must log on with the same user name and account that was used when the file was created.

```
:HELLO MEMBER1.PAYACCT
```

← Log on to system with user name and account of file creator.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:RENAME EMPLOYEE, EMP11
ERR 120
CREATOR CONFLICT
:RENAME EDITFILE, CUSTOMER
:LISTF
```

← Request to change name of EMPLOYEE file to EMP11. System prints error since MEMBER1 did not create the file named EMPLOYEE.

← Request to change name of EDITFILE to CUSTOMER. Change is successful since the file was created by MEMBER1.

```
FILENAME

CUSTOMER    EMPFILE     EMPLOYEE    FILEONE     FILETEM     FILETWO
FILEXXX     XFILE
```

In order to change the name, you must be the creator and also have S (save) access to the group to which the file belongs and W (write) access to the file. You can log on with the group name, use the group name to qualify the file name, or the group can be your home group.

If the file to be renamed has a lockword assigned, you must provide it with the name or when the system prompts for it. (See the next example.)

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

assigning, changing, and removing a lockword

The :RENAME command is also used to assign, change, or remove a disc file's lockword.

```
:RENAME CUSTOMER, CUSTOMER/LOCKA
:
```

Assign the lockword LOCKA to the CUSTOMER file.

A lockword may be from 1 to 8 alphanumeric characters; the first character must be alphabetic.

```
:RENAME FILEONE, FILEONE/REDROCK
:RENAME FILEONE, FILEONE
LOCKWORD: FILEONE.GROUP1.PAYACCT?
REDROCK
```

Assign lockword of REDROCK. If you try to remove the lockword, the system prompts for the current one.

As you can see from this example, to remove a lockword, you rename the file using its name and the current lockword as the first parameter and its name without the lockword for the second parameter.

To change the lockword, use the file name and a new lockword for the second parameter.

```
:RENAME CUSTOMER/LOCKA, CUSTOMER/KEY4LOCK
:
```

Change CUSTOMER file's lockword to KEY4LOCK.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

copying data to and from devices

copying data from cards to a disc file

To copy a file stored on punched cards to a disc file, you can use the FCOPY utility.

```
:HELLO MEMBER1.PAYACCT
```

Log on to the system.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

Specify the characteristics of the disc file to be created.

```
:FILE DISCFILE,NEW; REC=-80,3,F,ASCII; SAVE; DISC=200,2
:FILE CARDS; REC=,,,ASCII; DEV=CARD
:RUN FCOPY.PUB.SYS
```

Define CARDS as a card reader device file with ASCII formatted data.

Initiate FCOPY execution.

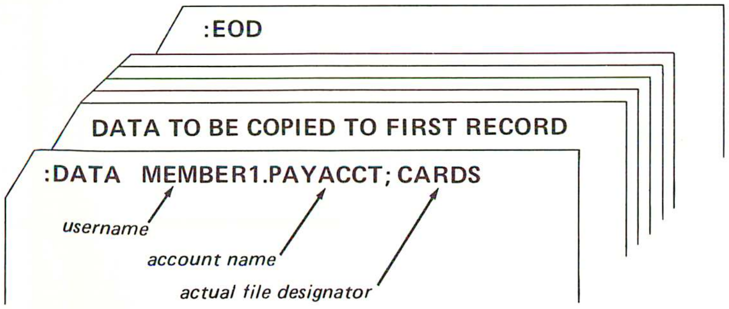
```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=*CARDS; TO=*DISCFILE
EOF FOUND IN FROMFILE AFTER RECORD 27

28 RECORDS PROCESSED *** 0 ERRORS
```

Copy from the file named CARDS to the file named DISCFILE. Use asterisks to indicate a :FILE command has been entered to define these files.

```
>EXIT

 END OF PROGRAM
```

After cards are copied, terminate FCOPY.

The cards that you copy should be preceded by a :DATA command card and followed by an :EOD card.


:EOD
DATA TO BE COPIED TO FIRST RECORD
:DATA MEMBER1.PAYACCT; CARDS
username
account name
actual file designator

The format of the :DATA command is:

:DATA [*jobname,*]*username* [/*upass*].*acctname* [/*apass*] [;*filename*]

(Parameters enclosed in brackets are optional.)

The *username* and *acctname* parameters must exactly match the log-on username and account name used in the :HELLO or :JOB command. If you include a filename, you must access the file with that same name. If you don't specify a name you will access the first spooled file belonging to your username and account.

After you place the cards in the card reader hopper and turn the power on, press the RESET button. If the card reader is a spooled device, the cards are read in immediately and written to a special disc file until FCOPY accesses the CARDS file. If the card reader is not a spooled device, the :DATA command card is read but the rest of the cards remain in the hopper until the FCOPY program accesses the CARDS file.

You can prepare the card reader and read the :DATA card either before or after you initiate your session. The :DATA command card simplifies card reader operations since the console operator does not have to intervene and allocate a device with device class CARD.

The disc file is a new file named DISCFILE with the following characteristics:

• 80 character records	REC=-80,3,F,ASCII;
• 3 records per block	
• fixed length records	
• ASCII format	
• permanent file	SAVE;
• maximum of 200 records	DISC=200,2
• 2 extents	

The system supervisor defines the device class for the card reader when the system is configured. In this example, CARD is the device class name.

The format for a card reader device file is BINARY by default. If you include the REC=,,,ASCII parameter in the :FILE command, you can avoid getting this warning message:

```
>FROM = *CARDS; TO = *DISCFILE
*201*;
WARNING: FROMFILE IS BINARY, TOFILE IS ASCII
CONTINUE OPERATION (Y OR N) ?Y
```

If you do get the message, respond with a "Y" and the data will be copied successfully.

File names preceded by an asterisk *back-reference* a :FILE command. In other words, the asterisk tells the system to use the information from a :FILE command that has been entered earlier in the session or job with the same formal file designator that follows the asterisk. If no :FILE command is found, an error occurs.

Another method for copying card data to a disc file uses the NEW parameter of the FCOPY program.

```
:FILE CARDS; REC=,,,ASCII; DEV=CARD
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=*CARDS; TO=DISCF; NEW
EOF FOUND IN FROMFILE AFTER RECORD 8

9 RECORDS PROCESSED *** 0 ERRORS
```

Since the card reader has undefined length records, the disc file will also have undefined length records if you use this method. FCOPY creates the TO-file with the FROM-file characteristics unless you use a :FILE command to define characteristics for the TO-file.

If you use this method the new file is created with 1023 records and, therefore, the maximum number of cards that can be copied is 1023. If you want to copy more than 1023 cards, you should create a file with the :BUILD command first and then copy the cards to that file.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

copying data from tape to a disc file

To copy a magnetic tape file to a disc file, use the FCOPY utility.

```
                                                 170  1
                             7000
:BUILD FILEXYZ/LYXZ; DISC=~~500~~,4; REC=-~~80~~,~~3~~,F,ASCII      ⟸ Create disc file.
:FILE TAPFILE; DEV=TAPE; REC=-~~80~~,,F,ASCII              ⟸ Specify device type for TAPFILE.
:RUN FCOPY.PUB.SYS                -170                     ⟸ Initiate FCOPY execution.

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=*TAPFILE; TO=FILEXYZ/LYXZ; SUBSET
EOF FOUND IN FROMFILE AFTER RECORD 27

28 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
```

⟸ Tell FCOPY to copy one file from the magnetic tape device referenced by TAPFILE to the disc file FILEXYZ with lockword LYXZ.

In this example, FILEXYZ is created before FCOPY is executed so that the disc file is bigger than the tape file and additional records can be added later. When using a :BUILD or :FILE command instead of the FCOPY ;NEW option, you can specify other file characteristics such as the number of extents.

The :FILE command that defines TAPFILE includes the REC parameter to specify that the tape records are 80 characters in length and the data is in ASCII format. If the parameter group is not included, you will be given two warnings by FCOPY.

```
*200*;                                       ⟸ Reply with printable character.
WARNING: FROMFILE RECSIZE IS 128 WORDS, TOFILE RECSIZE IS 80 BYTES.
CONTINUE OPERATION (Y OR N) ?Y               ⟸ FCOPY prints warning. Reply with "Y."
*201*L                                       ⟸ Reply with printable character.
WARNING: FROMFILE IS BINARY, TOFILE IS ASCII
CONTINUE OPERATION (Y OR N) ?Y               ⟸ Reply with "Y" to continue.
```

These warnings appear because you have defined a record size of 80 characters for the disc file and specified that it is to contain ASCII data. The FROM-file records are truncated in this case. If the FROM-file records are shorter than the TO-file records, the contents of the extra character positions are unpredictable.

When copying from a tape, you must specify SUBSET to indicate that you only want to copy to the first end-of-file mark. If you do not include this parameter, FCOPY stops when a parity error occurs due to the end of all data on the tape.

If the file has a lockword, as it does in the example, you can either supply a slash and the lockword following the file name or MPE will prompt for the lockword. For example:

```
   LOCKWORD: FILEXYZ.GROUP1.PAYACCT?
```

When FCOPY opens the tape file, a message is printed on the system console:

device class name

```
?IO/10:27/#S1062/42/LDEV# FOR "TAPFILE" ON TAPE (NUM)
```

time *session number* *process identification number (PIN)* *formal file designator*

To respond to this message, press the control key and A simultaneously. The system responds with an equals (=) sign. Enter the following response:

```
=REPLY 42,7
```

logical device on which tape is mounted (device 7 in this example)

process identification number (PIN)

FCOPY then copies the tape and rewinds it when it closes the tape file.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

copying data from a disc file to the line printer or a terminal

The FCOPY utility can also be used to copy data from a disc file to the line printer or to your terminal.

```
:FILE PRINTER; DEV=LP
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM =FILEXYZ/LYXZ; TO=*PRINTER
*200*
EOF FOUND IN FROMFILE AFTER RECORD 9

10 RECORDS PROCESSED *** 0 ERRORS
```

Define a devicefile named PRINTER with device class LP (line printer).

Initiate FCOPY execution.

Specify FILEXYZ with lockword LYXZ as the FROM-file and back-reference the PRINTER file :FILE command.

Press return to ignore warning.

10 records are copied to the line printer.

The line printer output looks like this:

```
527-58-31101BOWERS     SUSAN A   13 OAKWOOD DRIVE SC FSHF  400 1    0YY
678-99-33444PARK       RICHARD R 1290 FRIAR WAY   CU MSHF  400 1    0YY
089-23-51023OSGOOD     JOSEPH K  4567 CALIF. ST.  MV MSHF  450 3    0YY
432-11-92012HOUGHTON   MARGE S   313 MOONBEAM WAY LA FSHF  450 1    0NY
275-86-38275SPIEGLE    ROSALIE I 22 N. 3RD STREET SJ FSSF35000 0  750YY
138-66-09021RICHARDS   NORMAN A  1961 MULBERRY DR.LG MMSF35000 4    0NY
499-73-82472PIERCE     EMMA R    460 SARATOGA AVE.SCLFMSF20000 2    0YY
253-48-00033PAPADAKIS  JAMES L   1347 KINGSWORTH  CU MMSF20000 3    0YY
537-58-23454MAC LEAN   VINCENT T 819 PALO ALTO ST.MV MMSF15000 5    0YY
333-44-82825KONIGSBERG JOANNE M  45 ELDEN STREET  LA FMSF15000 2    0NY
```

If the records are longer than 132 characters, you must include the CHAR parameter or the entire record will not be copied to the line printer. In this case, you should also use the NORECNUM parameter to suppress the file identification, record numbers, and word offset numbers.

If you are copying a disc file to the line printer in job mode, you do not need to use the :FILE command or a TO-file. This is because the default TO-file is $STDLIST and, in job mode, $STDLIST is the line printer.

In the next example, the first 5 records are copied to the terminal from the disc file EMPLOYEE.

Omit the TO-file to send output to your terminal. Use the SUBSET parameter.

```
>FROM=EMPLOYEE; TO=; SUBSET=0:4
*200*;
WARNING: FROMFILE RECSIZE IS 108 BYTES, TOFILE RECSIZE IS 80 BYTES.
CONTINUE OPERATION (Y OR N) ?Y
678-99-33444PARK        RICHARD R  1290 FRIAR WAY   CU MSHF   400 1    0YY
089-23-51023OSGOOD      JOSEPH K   4567 CALIF. ST.  MV MSHF   450 3    0YY
432-11-92012HOUGHTON    MARGE S    313 MOONBEAM WAY LA FSHF   450 1    0NY
275-86-38275SPIEGLE     ROSALIE I  22 N. 3RD STREET SJ FSSF35000 0  750YY
138-66-09021RICHARDS    NORMAN A   1961 MULBERRY DR.LG MMSF35000 4    0NY
5 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
```

The TO-file is assumed to be the standard job or session list device, $STDLIST, if you do not specify a file name for it. $STDLIST is your terminal if you are operating in session mode. The record size for a terminal is 80 bytes (characters) by default. FCOPY warns you that the FROM-file and the TO-file are not the same size.* You can ignore the warning by entering a "Y" in response to the prompt that follows the warning message (or pressing carriage return when *200* is displayed).

*If you want to have the warning message printed, enter any printing character except a colon (:). To skip printing the message, press the carriage return.

You may specify a subset of the file as

SUBSET=starting-record-number : last-record-number

or

SUBSET=starting-record-number , number-of-records

as well as several other ways described in the *FCOPY Reference Manual.*

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

printing information on the line printer

Many MPE commands allow you to specify where the command's output is to be printed. The parameter you use to specify this is usually called the *listfile* parameter. To use the line printer as the *listfile* when you are operating in session mode, define the line printer as a devicefile by using the :FILE command and then use the back-reference form of the file name as the *listfile* parameter. For example, to list information about files on the line printer:

```
:HELLO MEMBER1.PAYACCT
```

Log on to system.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:FILE LP;DEV=LP
:LISTF @.GROUP2,2; *LP
```

Define a file, in this case named LP, with device class LP (line printer). Use the file name preceded by an asterisk (*) as the *listfile* parameter.

In this example, the @ symbol and group name are used with the LISTF command to request information about all the files in GROUP2. The second parameter, 2, specifies the type of information to be listed. Here is an example of the information as it is displayed on the line printer:

```
ACCOUNT=  PAYACCT      GROUP=  GROUP2

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

DISCFILE          80B  FA          28        200   3        34  1   2 ???
DISCFL            80B  FA          20       1023   3        43  1   8 ???
FILEX             80B  FA          20         40   3        15  1   1 ???
FILEXYZ           80B  FA          28        500   3        42  1   4 ???
```

The file name you use to define the devicefile is arbitrary as long as it is 8 alphanumeric characters or less, beginning with an alphabetic character. For example, you can use the names PRINTER, L, LINEPRINTER, PR, or X.

If the line printer is a spooled device, you may want to specify the output priority for the listing and the number of copies to be made. For example:

```
:FILE PRINTER;  DEV=LP,13,4 <------ numcopies
                        /
         outputpriority
```

specifies that the listing is to have the highest priority and that four copies are to be made.

This form of the :FILE command is often used in production jobs to specify the number of copies of a report to be printed. The name of the output file in the program can be used directly

```
:FILE OUT;  DEV=LP,13,4
```

or it can be equated to a different device file name.

```
:FILE OUT=PRINTER;  DEV=LP,13,4
```

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

copying, combining, and transferring files

combining files

The FCOPY utility program can be used to combine two or more files by using an asterisk (*) as the TO-file name.

```
:FILE MAGTAPE; DEV=TAPE; REC=-80,9,F,ASCII
:RUN FCOPY.PUB.SYS
```

Specify file MAGTAPE as a magnetic tape with 80 character records, 9 records per block.

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM = EMP1; TO = *MAGTAPE
EOF FOUND IN FROMFILE AFTER RECORD 9

10 RECORDS PROCESSED *** 0 ERRORS
```

Copy from disc file EMP1 to the beginning of the tape. (Asterisk back-references the :FILE command).

```
>FROM=EMP2; TO=*
EOF FOUND IN FROMFILE AFTER RECORD 9

10 RECORDS PROCESSED *** 0 ERRORS
```

Copy from EMP2 disc file to the tape beginning at the current position following the EMP1 data. (Asterisk indicates copy from current output tape position.)

```
>FROM=EMPLOYEE; TO=*
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS
```

Copy from EMPLOYEE disc file to the tape beginning at the current position following the EMP2 data.

```
>EXIT

 END OF PROGRAM
```

Terminate FCOPY program.

Only one end-of-file mark is written to the output tape and it follows the last record copied.

When FCOPY opens the tape file, a message is printed on the system console:

```
?IO/12:20/#S1061/29/LDEV# FOR "MAGTAPE" ON TAPE (NUM)
=REPLY 29,8
```

process identification number

Control A causes MPE to print = sign

Respond with Control A, REPLY, the process identification number (29), and the logical device number of the tape device (8 in this example).

FCOPY prints a warning if the records of the TO-file are not the same size as the records of the FROM-file. If you indicate you want to continue running FCOPY, the records are copied. (See "Copying Data from Tape to a Disc File" in this manual for details).

If instead of using an asterisk you repeat the TO-file name in an FCOPY command following the first one, the TO-file is rewound and the tape device is set to an offline (or "not ready") state. For example, if the second FCOPY command in the example above is changed to

```
>FROM=EMP2;TO=*MAGTAPE
```

the tape rewinds and the tape device is no longer in a ready state. The system prints a message such as this on the system console:

```
IO/12:21/LDEV#8 NOT READY
```

You must then begin again by copying the first file to the tape.

You can also combine files by copying to another disc file using the :BUILD command to create a file large enough to contain the combined files and then using the same techniques as illustrated in this example to copy the files.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

combining ASCII disc files with the editor

If you want to combine files containing ASCII data, it may be easier to use the Editor than FCOPY.

```
:EDITOR

HP32201A.5.01 EDIT/3000   FRI, JAN 21, 1977,   3:40 PM
(C) HEWLETT-PACKARD CO. 1976
/TEXT EMP1,UNN
/JOINQ EMP2,UNN
NUMBER OF LINES JOINED = 10
/KEEP EMPALL,UNN
/EXIT
IF IT IS OK TO CLEAR RESPOND "YES"
CLEAR? Y

 END OF SUBSYSTEM
```

← Initiate Editor execution.

← Move the contents of the EMP1 disc file into the work file. Then move the contents of the EMP2 disc file into the work file following the EMP1 data. Save the combined files in a new file named EMPALL.

The line or record length is established by the file specified in the TEXT command. If the records that are added with the JOIN command are longer than the records that are already in the work file, the records are truncated to the proper size. If the file that is added with the JOIN command is shorter, the records are padded with blanks.

If you use the JOINQ form of the JOIN command, as shown in this example, the records are not displayed on the terminal when they are joined to the work file.

Here are some important things to remember about the file to be joined to the work file:

- Only disc files can be joined to a work file. If you want to join a tape file, you must first use FCOPY to copy the tape file to a disc file.
- The file code must be a number between 0 and 1023. In most cases, it will be the system default of 0. You can use LISTF to see what the file code is.
- The file data must be in ASCII format.
- The JOIN command appends a file's records to the WORK file unless instructed to insert them at a particular line number. If the JOIN file is unnumbered and you do not specify UNN, the last eight characters of each record will be discarded.

Before using the TEXT command, you may want to use the Editor SET SIZE= command to make the work file large enough to contain the combined files. Otherwise you may get the message SCRATCH FILE IS FULL. KEEP, THEN TEXT AGAIN. Save the current work file contents and then TEXT the saved file in again.

ADDITIONAL INFORMATION:

EDIT/3000 Reference Manual

copying a subset of a file to another file

In this example, all hourly employees in the EMPLOYEE file are selected and copied to a new file named HOURLY that is created by FCOPY.

```
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=EMPLOYEE; TO = HOURLY; NEW; SUBSET="H",58
EOF FOUND IN FROMFILE AFTER RECORD 19

9 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
```

Initiate FCOPY execution.

Copy all EMPLOYEE records with an "H" in character position 58 to a new disc file named HOURLY.

Nine records are located and copied to the HOURLY file.

If you want to copy the records of all employees who live in Santa Clara to the line printer, use a :FILE command to define the TO-file.

```
:FILE PRINTER; DEV=LP
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=EMPLOYEE; TO=*PRINTER; SUBSET="SCL",53
*200*L
WARNING: FROMFILE RECSIZE IS 80 BYTES, TOFILE RECSIZE IS 132 BYTES.
CONTINUE OPERATION (Y OR N) ?Y
EOF FOUND IN FROMFILE AFTER RECORD 19

3 RECORDS PROCESSED *** 0 ERRORS

>EXIT
```

Define PRINTER as a devicefile with device class LP and initiate FCOPY execution.

Copy from EMPLOYEE file to line printer all records with SCL beginning in character position 53.

FCOPY warns that the line printer records are longer than the EMPLOYEE records. Reply with a Y to ignore the warning.

The asterisk preceding PRINTER indicates that PRINTER has been defined with a :FILE command.

The line printer listing looks like this:

```
499-73-82472PIERCE       EMMA R     460 SARATOGA AVE.SCLFMSF20000 2    0YY
456-97-00543GOMEZ        ROBERT D   1331 POETT LANE  SCLMMSF27000 2    0YY
300-24-36664JUNG         DIANA T    965 WINCHESTER   SCLFMHP  850 2    0YY
```

ADDITIONAL INFORMATION:

FCOPY Reference Manual

copying a disc file from one group to another

You can copy a file from a group to which you have R (read) access to another group to which you have S (save) access by qualifying the FROM-file name with its group name. Suppose the account manager creates a group named GROUPX with these file access capabilities: (R:AC; A,W,L,X,S:GU). Every member of the PAYACCT account has read access to the files in this group. If MEMBERA whose home group is GROUP2 wants to copy a file from GROUPX to GROUPA, he does it as shown in the following example.

```
:HELLO MEMBERA.PAYACCT
```

MEMBERA logs on to PAYACCT and his home group GROUP2.

```
:RUN FCOPY.PUB.SYS
```

MEMBER initiates FCOPY.

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=XFILEX.GROUPX; TO=XFILEX; NEW
EOF FOUND IN FROMFILE AFTER RECORD 27
```

Copy XFILEX from GROUPX to a new file in GROUP2 named XFILEX.

```
28 RECORDS PROCESSED *** 0 ERRORS
```

Note that although this message is printed, the new file is not actually saved until FCOPY closes the file and terminates.

```
>EXIT
```

If you need to copy a file to your group from a group to which you do not have R (read) access, you can do so provided the creator of the file releases the file's security provisions. Only the creator can do this. He identifies himself by logging on with the correct user name, account and group. The creator may log on to a group that does not contain the file and release its security provisions by qualifying the file name with the name of the group to which the file belongs, for example, DISCFILE.GROUP1.

```
:HELLO MEMBER1.PAYACCT
```

File creator logs on to PAYACCT (home group is GROUP1).

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:RELEASE DISCFILE
:HELLO MEMBERA.PAYACCT
```

Creator releases the security for DISCFILE. MEMBERA whose home group is GROUP2 logs on to PAYACCT.

End of MEMBER1's session message is displayed.

```
CPU=1.  CONNECT=1.  MON, JAN 9, 1978, 5:24 PM

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

Beginning of MEMBERA's session message is displayed.

```
:RUN FCOPY.PUB.SYS
```

MEMBERA initiates FCOPY.

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=DISCFILE.GROUP1; TO=FILEX; NEW
EOF FOUND IN FROMFILE AFTER RECORD 27
```

Copy DISCFILE from GROUP1 to a new file named FILEX and save it in log-on group, GROUP2.

```
28 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
:LISTF
```

If the files in GROUP2 are listed, the new file FILEX is in the group.

```
FILENAME

FILEX
```

```
:HELLO MEMBER1.PAYACCT
```

Creator logs on again.

(End of session and beginning of session messages are displayed.)

```
:SECURE DISCFILE
```

Creator reinstates DISCFILE's security provisions.

The file's security provisions need not be released if you log on to the group containing the file. To do so you must know the group name and its password, if one is assigned.

```
:HELLO MEMBERA.PAYACCT,GROUP1
            .
            .
            .
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=DISCFILE; TO=FILEX.GROUP2; NEW
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS

>EXIT
END OF PROGRAM
```

MEMBERA logs on to PAYACCT group GROUP1.

Initiate FCOPY.

Copy DISCFILE from GROUP1 to a new file named FILEX in GROUP2.

In this example, MEMBERA logs on to GROUP1 and can copy a file from GROUP1 to his home group, GROUP2.

In the next example, MEMBER1 attempts to copy a file to GROUP2 when logged on to his home group, GROUP1.

```
:HELLO MEMBER1.PAYACCT

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM


:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=FILEXYZ/LYXZ; TO=FILEXYZ.GROUP2;NEW
EOF FOUND IN FROMFILE AFTER RECORD 27

28 RECORDS PROCESSED *** 0 ERRORS

>EXIT
*104*L
CAN'T SAVE NEW TOFILE

DISPLAY FILE INFORMATION (Y OR N) ?N
```

MEMBER1 logs on to PAYACCT with home group GROUP1.

Copy FILEXYZ from GROUP1 to a new file with the same name in GROUP2.

Copy complete but FCOPY does not save new file until program is terminated.

Terminate FCOPY.

When FCOPY tries to create the new file it discovers that only group users can create files in GROUP2. The copy fails.

A response of N inhibits the printing of a file information display (FID) explaining error.

The results of this operation depend on the way GROUP2 is defined by the account manager. In this case, the group is defined so that only group users (GU) have save (S) access to the group.

If MEMBER1 logs on to GROUP2, he can successfully copy the file. Since GROUP1 is MEMBER1's home group, he need not release the security provisions.

```
:HELLO MEMBER1.PAYACCT,GROUP2
```

MEMBER1 logs on to PAYACCT group GROUP2.

```
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=FILEXYZ.GROUP1; TO=FILEXYZ;NEW
LOCKWORD: FILEXYZ.GROUP1.PAYACCT?
;
EOF FOUND IN FROMFILE AFTER RECORD 27

28 RECORDS PROCESSED *** 0 ERRORS

>EXIT
```

Copy FILEXYZ from GROUP1 to a new file of the same name in GROUP2, the log-on group.

Since the file has a lockword, FCOPY requires that it be supplied.

In summary, when copying files from one group to another you must be aware of:

- who created the file
- what type of file access is defined for the group containing the file and the group to which the file is to be copied
- what groups you have access to when you log on.

If you log on to your home group you may be able to save files in that group only. If you log on to another group, and the groups are defined with the system defaults for file access capability, you can save files in both your home group and your log on group.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

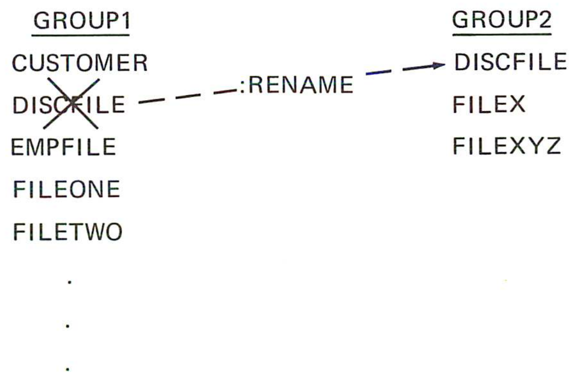
transferring a disc file from one group to another

To transfer a file from one group to another, use the :RENAME command. You must be the creator of the file in order to do this.

```
:HELLO MEMBER1.PAYACCT,GROUP2                Creator of file logs on to group to which file is to be moved.

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:RENAME DISCFILE.GROUP1, DISCFILE            DISCFILE is moved from GROUP1 to log-on group, GROUP2.
:LISTF

FILENAME

DISCFILE    FILEX                            DISCFILE is now in GROUP2 in the system directory.
```

You must have SAVE access to the group to which the file is to be transferred. Assuming GROUP2 was defined with the system default file access capability, only group users (GU) may save files in the group. Since MEMBER1's home group is GROUP1, he must log on to GROUP2 in order to save a file in that group.

Note that the file named DISCFILE is no longer in GROUP1 after the :RENAME command is executed. The file is transferred rather than copied.



ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

copying a disc file from one account to another

In order to copy a file from one account to another, the file's creator must agree to temporarily release the security provisions for the file (unless the account allows R (read) access to ANY system user).

```
:HELLO MEMBER1.PAYACCT

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

File creator logs on to PAYACCT and home group, GROUP1, that contains file.

```
:RELEASE DISCFILE
:HELLO USERA.PERSONL
```

Creator releases DISCFILE's security provisions.

User who wants to copy file to another account logs on to that account.

End of MEMBER1's session.

```
CPU=1.  CONNECT=3.  MON, JAN 9, 1978, 5:00 PM

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

Beginning of USERA's session.

```
:RUN FCOPY.PUB.SYS
```

Initiate FCOPY execution.

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=DISCFILE.GROUP1.PAYACCT; TO=FILEX; NEW
EOF FOUND IN FROMFILE AFTER RECORD 27

28 RECORDS PROCESSED *** 0 ERRORS
```

Copy DISCFILE from GROUP1 of PAYACCT to a new file named FILEX to be created and saved in USERA's home group of the PERSONL account.

```
>EXIT
```

FILEX is saved when FCOPY closes the file.

```
 END OF PROGRAM

:HELLO MEMBER1.PAYACCT
```

File creator logs on again.

(End of session and beginning of session messages are displayed.)

```
:SECURE DISCFILE
```

Creator reinstates file's security provisions.

If you want to move a file rather than duplicate it in another account, the creator can use the :PURGE command and delete the file from the account in which it was originally created instead of using the :SECURE command.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

copying an EBCDIC file to an ASCII disc file

If you have a tape file containing EBCDIC data, you can copy it to a disc file and translate the data to ASCII format at the same time by using FCOPY.

```
:FILE DISCFILE,NEW; REC=-80,3,F,ASCII; SAVE; DISC=2000,4
:FILE TFILE; DEV=TAPE; REC=-80,10,F,ASCII
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=*TFILE; TO=*DISCFL; SUBSET; EBCDICIN
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
:LISTF DISCFL,2
ACCOUNT=   PAYACCT      GROUP=   GROUP1

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

DISCFL              80B  FA           20       1023   3        43  1  8 ???
```

ASCII format

Specify a tape file named TFILE with 80 character records, 10 per block. ASCII indicates non-binary in this case.

Copy from the TFILE device to disc file named DISCFL. Stop at the first end-of-file mark. The input data is in EBCDIC format.

:LISTF displays the characteristics of the new file.

If you do not create the disc file with a :BUILD or :FILE. . . ,NEW command, the disc file will contain at most 1023 records by default.

The EBCDICIN parameter instructs FCOPY to translate the EBCDIC data to ASCII format.

If the tape you are copying has a label and you do not want to copy it, you can skip it by specifying:

```
>FROM=*TFILE; TO =*DISCFL; SUBSET; EBCDICIN;
   SKIPEOF=1
```

The SKIPEOF=1 parameter tells FCOPY to skip past the first end-of-file mark before copying any information.

If the input tape contains fields written in packed decimal format, you use FCOPY. Alternatively, you can write an application program that uses the system intrinsic CTRANSLATE to translate the EBCDIC fields and leave the packed decimal fields unchanged.

If you do not include the REC= parameters in the :FILE command for the tape file and you try to copy to a file that does not have 128 word records, FCOPY prints a warning. It prints another warning if you do not specify ASCII data and you try to copy to a file defined with ASCII formatted data. These warnings are printed because the FCOPY default record length for a tape file is 128 words and the default data format is BINARY.

```
:BUILD DISCF; REC=-80,3,F,ASCII; DISC=500,2
:FILE TFILE; DEV=TAPE
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=*TFILE; TO=DISCF; SUBSET; EBCDICIN=(1,5;6:10;21:30)
```

Create disc file named DISCF.

Define TFILE. Omit REC= parameters.

Copy from TFILE to DISCF.

See "Copying Data from Tape to a Disc File" in this manual for more information about FCOPY warning messages.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

copying an ASCII file to an EBCDIC file

FCOPY can create files containing EBCDIC formatted data.

```
:FILE TFILE; DEV=TAPE; REC=-80,10,F,ASCII
:RUN FCOPY.PUB.SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

 FROM=EMPLOYEE; TO=*TFILE; EBCDICOUT
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
```

← Define a tape file named TFILE with 80 character records in blocks of 10 containing ASCII (non-binary data).

← Copy from a disc file named EMPLOYEE containing ASCII data to the TFILE tape file, translating the data to EBCDIC format.

The EBCDICOUT parameter instructs FCOPY to translate the ASCII characters to EBCDIC format.

If the ASCII disc file contains packed decimal data, FCOPY does not copy the data correctly. Therefore, you should avoid using packed decimal data fields in a file that is to be translated to EBCDIC format with FCOPY. These data fields should be written in ASCII format and converted to packed decimal data format on the system to which the EBCDIC data is being transferred. (You can copy the tape with a program of your own that uses the CTRANSLATE intrinsic).

FCOPY also translates ASCII data to BCD (Binary coded decimal) format for systems requiring data in that format. In this case you specify BCDOUT.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
MPE Commands Reference Manual (Section VI)

sorting data

The SORT program may be run as a stand-alone program during either a session or a job. In this example, it is run from a terminal during a session.

```
:RUN SORT.PUB.SYS
```

Initiate SORT execution. SORT is in the PUB group of the SYS account.

```
HP32214B.01.02 SORT/3000  FRI, JAN 21, 1977, 10:59 AM
(C) HEWLETT-PACKARD CO. 1976

>INPUT EMPLOYEE
>OUTPUT SORTEMPL
>KEY 58,1
>KEY 13,12
>VERIFY
```

SORT prompts for commands with greater than (>) symbol.

Specify EMPLOYEE as the input file and SORTEMPL as the output file.

Specify the major sort key field by indicating the first character and length of the field. Then specify the second level sort key.

The VERIFY command requests a summary of the SORT options which are in effect.

```
INPUT FILE = EMPLOYEE
OUTPUT FILE = SORTEMPL
KEY POSITION    LENGTH       TYPE     ASC/DESC
       58          1         BYTE        ASC     (MAJOR KEY)
       13         12         BYTE        ASC

>END
```

ascending sequence

```
                         STATISTICS
```

A summary of statistics about the sort is printed after the sort is completed.

```
NUMBER OF RECORDS =                                  20
RECORD SIZE (IN BYTES) =                            108
NUMBER OF INTERMEDIATE PASSES =                       0
SPACE AVAILABLE (IN WORDS) =                     13,201
NUMBER OF COMPARES =                                 89
NUMBER OF SCRATCHFILE IO'S =                         18
CPU TIME (MINUTES) =                                .01
ELAPSED TIME (MINUTES) =                            .01

 END OF PROGRAM
```

When you specify a key, you can define the type of data to be sorted and the sequence in which it is to be sorted. For example:

```
>KEY 58,1,BYTE,ASC
```

type *sequence*

If these parameters are omitted, as they are in the example, the defaults are BYTE (character) and ASC (ascending sequence).

BYTE type keys are sorted byte by byte according to the value of the 8 bits in each byte. If the data is ASCII it will be in ASCII collating sequence after the sort. If the data is EBCDIC, it will be in EBCDIC collating sequence.

Refer to the example on the preceding page. Records of the EMPLOYEE disc file are sorted according to the information in the 1-character field in character position 58.

The second level of sorting is based on 12 characters beginning with character position 13.

The sorted data is written to a new disc file created by SORT and saved with the name SORTEMPL. The characteristics of the new file are the same as the characteristics of the input file EMPLOYEE.

Sorted records can be sent to the line printer device file. Note that the back reference indicator, the asterisk, is not required in this case.

```
:FILE LP; DEV=LP
:RUN SORT.PUB.SYS
```

Define a device file named LP with device class LP and initiate SORT execution.

```
HP32214B.01.02 SORT/3000  FRI, JAN 21, 1977, 11:07 AM
(C) HEWLETT-PACKARD CO. 1976

>INPUT EMPLOYEE
>OUTPUT LP
>KEY 13,12,DESC
>END
```

Specify the input file, EMPLOYEE.
Specify the output file, LP (the line printer).
Specify the major sort key: 12 characters beginning in character position 13 to be sorted in descending sequence.

```
                    STATISTICS

NUMBER OF RECORDS =                         20

                        .
                        .
                        .
```

The sorted records are printed on the line printer:

```
322-11-35874YEE         JANICE K   310 BEEKMAN PLACECU FSSF27000 0    0YY
275-86-38275SPIEGLE     ROSALIE I  22 N. 3RD STREET SJ FSSF35000 0  750YY
589-24-21124SCRUGGS     RODNEY X   76 AUBURN DRIVE  CU MMHP   800 1    0YY
138-66-09021RICHARDS    NORMAN A   1961 MULBERRY DR.LG MMSF35000 4    0NY
342-48-83295POWELL      PATRICIA J 249 FREMONT AVE. LA FMHF  1000 4    0YY
499-73-82472PIERCE      EMMA R     460 SARATOGA AVE.SCLFMSF20000 2    0YY
678-99-33444PARK        RICHARD R  1290 FRIAR WAY   CU MSHF   400 1    0YY
253-48-00033PAPADAKIS   JAMES L    1347 KINGSWORTH  CU MMSF20000 3    0YY
089-23-51023OSGOOD      JOSEPH K   4567 CALIF. ST.  MV MSHF   450 3    0YY
466-39-44334OROZCO      JOSEPH N   112 BRIGHTEN AVE.MV MMHP   800 2    0YY
537-58-23454MAC LEAN    VINCENT T  819 PALO ALTO ST.MV MMSF15000 5    0YY
333-44-82825KONIGSBERG  JOANNE M   45 ELDEN STREET  LA FMSF15000 2    0NY
300-24-36664JUNG        DIANA T    965 WINCHESTER   SCLFMHP   850 2    0YY
244-75-38445JENSEN      HILDA A    17650 SHADOW LANELG FMHF  1000 0  800YY
432-11-92012HOUGHTON    MARGE S    313 MOONBEAM WAY LA FSHF   450 1    0NY
531-66-77771HORN        MICHAEL S  2230 BRONSON AVE.SJ MMSF22000 1    0NY
456-97-00543GOMEZ       ROBERT D   1331 POETT LANE  SCLMMSF27000 2    0YY
212-83-95432FERGUSON    KATHERINE L729 UNION AVENUE LG FMSF22000 0    0YY
527-58-31101BOWERS      SUSAN A    13 OAKWOOD DRIVE SC FSHF   400 1    0YY
142-67-55335ALEXANDER   GEOFFREY N 702 CORVALLIS DR.SCRMSSF45000 01000YN
```

character position 13

You may also call SORT procedures from your programs. Read about this in the SORT/3000 Reference Manual.

ADDITIONAL INFORMATION:

SORT/3000 Reference Manual (Sections I and II)
MPE Commands Reference Manual (Section VI)

using files in jobs

preparing a production job

The easiest way to prepare job control commands is to enter them in a disc file using the Editor (although they may also be punched in cards and read from the card reader). When you want to run a job stored in a disc file, you can use the :STREAM command as illustrated in the example following this one.

```
:EDITOR

HP32201A.5.01 EDIT/3000  TUE, JAN 25, 1977,  1:09 PM
(C) HEWLETT-PACKARD CO. 1976
/ADD
    1  ①→ !JOB XX34,MEMBER1.PAYACCT,GROUP1
    2        !COMMENT
    3   ②    !COMMENT -----------------------------------
    4        !COMMENT      ACCOUNTS RECEIVABLE UPDATE
    5        !COMMENT -----------------------------------
    6        !TELLOP *****************************************
    7        !TELLOP *         JOB XX34 STARTING             *
    8        !TELLOP *                                       *
    9   ③    !TELLOP *     ACCOUNTS RECEIVABLE UPDATE RUN -  *
   10        !TELLOP *     MOUNT 'CHKRCVD' TAPE AND STANDBY  *
   11        !TELLOP *     TO REPLY TO IT.                   *
   12        !TELLOP *                                       *
   13   ④    !TELLOP *****************************************
   14     →  !PURGE DCHKRD
```

:EDITOR — Initiate Editor execution.

/ADD — Add new lines to the work file.

① The first record is the JOB command with exclamation point (!) instead of colon (:).

(Example continued on next page.)

When preparing a job to be streamed, you must substitute another character for the colon normally used with MPE commands. Any ASCII character may be used except the letters A through Z or the numbers 0 through 9. The character most commonly used is the exclamation point for reasons you will see in the STREAM example which follows.

① The first command in a job must be the JOB command, followed by the user name, account name, and optionally the group name. In production work it is useful to specify a job name before the user name to identify the specific job that is being run. In the example, the job name is XX34.

② By using COMMENT commands, you can describe the function of the job within the job file.

③ The TELLOP command prints a message on the system console. In this example, the operator is notified that a job is starting and told to mount a particular tape for the job. Thus, if you initiate the job at a terminal, you can communicate with the operator.

④ To ensure that a DCHKRD file can be created, the PURGE command is used to delete an old DCHKRD file if one exists.

```
 15 (5)→ !FILE IN; DEV=TAPE; REC=40,3,F,ASCII                    ← Define input file.
 16     → !RUN FCOPY.PUB.SYS
 17 (6)   FROM=*IN; TO=DCHKRD; NEW; EBCDICIN; SUBSET; SKIPEOF=1
 18 (7) { EXIT                                                   ← Enter FCOPY commands in job.
 19       !EOD
 20       !TELLOP ** JOBXX34 FCOPY COMPLETE **
 21 (8) { !PURGE WORKFL                                          ← Enter commands to prepare files and
 22       !BUILD WORKFL; DISC=1000; REC=-80,3,F,ASCII              run application program named ACRUPD.
 23 (9)→ !FILE NEWCHK=DCHKRD
 24 (10)→ !FILE OUT=CUMACRC; ACC=APPEND
 25     → !RUN ACRUPD
 26 (11)  !TELLOP  ***********************                        ← Enter command to notify operator that job is finished.
 27 (12){ !TELLOP  *    JOB XX34   END    *
 28       !TELLOP  ***********************
 29       !EOJ                                                   ← Enter end of job command.
 30       ...                                                    ← Simultaneously press Control key and Y to stop adding lines.
/K ACRJOB,UNN                                                    ← Save job control commands in file named ACRJOB.
/E                                                               ← Terminate Editor.
```

(5) In this example, the IN file is defined as a tape device with 40 word (or 80 character) records in ASCII format, three per block, and fixed length.

(6) The first program to be run in the job is FCOPY.

(7) The FCOPY commands must be included since this is a job and not an interactive session. The FCOPY prompt character, >, is not included. FCOPY is instructed to skip to the first end-of-file mark (following an existing tape label), translate EBCDIC data on the tape to ASCII data, and copy it to a new disc file named DCHKRD. After this operation is finished, FCOPY is to terminate. The EOD command ensures that the program cannot read beyond this point even if something goes wrong. The TELLOP command notifies the console operator that the tape copy is complete.

(8) A file named WORKFL is used each time the job is run. The PURGE command ensures that a file of that name does not exist in the group. Each time the job is run, the old WORKFL is purged. If the file has already been purged, the message FILE NOT FOUND is printed. A new WORKFL is then created with the BUILD command. As a precaution in case the job does not run properly, the WORKFL is a permanent disc file rather than a temporary file.

(9) A file referenced as NEWCHK in the program is equated to an actual disc file named DCHKRD.

(10) A file referenced as OUT in the program is equated to a file name CUMACRC which is a permanent disc file. The ACC= APPEND parameter specifies that data is to be added following the existing data in the file. If this parameter is omitted and data is written in sequential access mode, data will be written to the beginning of the file.

(11) The ACRUPD program is the second program in the job to be executed.

(12) The operator is notified that the job is complete. The operator also receives a system message to this effect but a more specific message is sometimes helpful.

If this job is punched in cards it looks like this:

```
                                                              :EOJ
                                                            :TELLOP **********************
                                                          :TELLOP *    JOB XX34 END      *
                                                        :TELLOP ***********************
                                                      :RUN ACRUPD
                                                    :FILE OUT; DEV=LP,13,3; CCTL
                                                  :FILE CUMACRC=DCHKRD; ACC=APPEND
                                                :BUILD WORKFL; DISC=1000; REC=-80,3,F,ASCII
                                              :PURGE WORKFL
                                            :TELLOP ** JOB XX34 FCOPY COMPLETE**
                                          :EOD
                                        EXIT
                                      FROM=*IN;TO=DCHKRD;NEW; EBCDICIN;SUBSET; SKIPEOF=1
                                    :RUN FCOPY.PUB.SYS
                                  :FILE IN; DEV=TAPE; REC=40,3,F,ASCII
                                :PURGE DCHKRD
                              :TELLOP ***********************************************
                            :TELLOP *
                          :TELLOP *     TO REPLY TO IT.
                        :TELLOP *            MOUNT 'CHKRCVD' TAPE AND STANDBY
                      :TELLOP *            ACCOUNTS RECEIVABLE UPDATE RUN -
                    :TELLOP*
                  :TELLOP *                   JOB XX34  STARTING
                :TELLOP *
              :TELLOP ***********************************************************
            :COMMENT -------------------------------------
          :COMMENT     ACCOUNTS RECEIVABLE UPDATE
        :COMMENT ----------------------------
      :COMMENT
:JOB XX34,MEMBER1.PAYACCT,GROUP1
```

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Sections IV and VI)
EDIT/3000 Reference Manual
FCOPY Reference Manual

executing job control commands stored in a file

You can initiate a job that is stored in a file by logging on in session mode and using the :STREAM command.

```
:HELLO MEMBER1.PAYACCT                      ← Initiate a session.

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:STREAM ACRJOB                              ← Initiate job stored in ACRJOB file.
 #J35
:
```

You can log on with the same user name, account, and group used in the :JOB command of the job you want to stream or with some other user, account, or group names as long as you have access to the file in which the job is stored.

If you use the exclamation point (!) as a substitute character for the colon (:), you only need to specify the input file containing the job. However, if you created the job with some other character, that character must be included with the :STREAM command. For example, if the job commands are preceded by an at-sign (@), the :STREAM command must look like this:

:STREAM ACRJOB,@

to notify MPE that the substitute character is @. If you provide no character, MPE assumes you have used the default (the exclamation point).

The :STREAM command is useful if you want to initiate a job in the midst of a session and continue operating in session mode. The system spools the job to a disc file, schedules the job to begin as soon as possible, and prompts for the next command in your session.

When the job begins execution, the following message is printed on the system console:

/17:53/#J35/LOGON FOR:MEMBER1.PAYACCT ON LDEV#5

job number

time

The :TELLOP commands display the following message on the console:

```
17:53/#J35/*********************************************************
17:53/#J35/*         JOB XX34   STARTING                            *
17:53/#J35/*                                                        *
17:53/#J35/*      ACCOUNTS RECEIVABLE UPDATE RUN -                  *
17:53/#J35/*      MOUNT 'CHKRCVD' TAPE AND STANDBY                  *
17:53/#J35/*      TO REPLY TO IT.                                   *
17:53/#J35/*                                                        *
17:53/#J35/*********************************************************
```

When the IN file is opened by FCOPY, this message appears on the console:

```
?IO/17:53/#J35/25/LDEV# FOR "IN" ON TAPE (NUM)
```

to which you or the console operator must respond (after the tape is mounted and the device is in the online state):

```
=REPLY 25,8
```

Press control A, type REPLY, type the process identification number (25), and the logical device number of the tape unit (8).

When the last four commands are executed, these messages appear on the console:

```
MS/17:55/#J35/***********************
MS/17:55/#J35/*    JOB XX34  END    *
MS/17:55/#J35/***********************
ST/17:55/#J35/LOGOFF
```

The job is listed on the line printer or $STDLIST device. For example:

```
:JOB XX34,MEMBER1.PAYACCT, GROUP1
 PRI= DS;  INPRI= 13;  TIME= ?
 JOB NUMBER = #J35
 THU, JAN 27, 1977,  5:53 PM
 HP32002A.00.05

:COMMENT
:COMMENT ---------------------------------------
:COMMENT     ACCOUNTS RECEIVABLE UPDATE
:COMMENT ---------------------------------------
:TELLOP  ***********************************************
:TELLOP  *        JOB XX34  STARTING                   *
:TELLOP  *                                             *
:TELLOP  *    ACCOUNTS RECEIVABLE UPDATE RUN -         *
:TELLOP  *    MOUNT 'CHKRCVD' TAPE AND STANDBY         *
:TELLOP  *    TO REPLY TO IT.                          *
:TELLOP  *                                             *
:TELLOP  ***********************************************
:PURGE DCHKRD
;FILE IN; DEV=TAPE; REC=40,3,F,ASCII
:RUN FCOPY.PUB.SYS
```

```
PAGE 1  HEWLETT-PACKARD 32212A.1.03 FILE COPIER  THU, JAN ...

FROM=*IN; TO=DCHKRD; NEW; EBCDICIN; SUBSET; SKIPEOF=1
EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS

EXIT

 END OF PROGRAM
:TELLOP ** JOBXX34 FCOPY COMPLETE **
:PURGE WORKFL
FILE NOT FOUND
:BUILD WORKFL; DISC=1000; REC=-80,3,F,ASCII
:FILE NEWCHK=DCHKRD
:FILE OUT=CUMACRC; ACC=APPEND
:RUN ACRUPD

RECORDS ADDED (COMPANY AND AMOUNT)
JONES MFG., INC.        $115.35
ACME WIDGET             $ 43.67
X & R SURPLUS           $489.01

 END OF PROGRAM
:TELLOP  **********************
:TELLOP  *   JOB XX34  END    *
:TELLOP  **********************
:EOJ
```

You can check the status of your job by entering the :SHOWJOB command.

```
:SHOWJOB

JOBNUM   STATE IPRI JIN  JLIST    INTRODUCED  JOB NAME

#S585    EXEC        53   53       THU   5:53P MEMBER1.PAYACCT
#J35     EXEC QUIET   5S LP        THU   5:53P MEMBER1.PAYACCT
#S576    EXEC        33   33       THU   4:20P EEC.HUSEBY
#S510    EXEC        44   44       THU 12:24P GINNY.SMITH
#S546    EXEC        34   34       THU   2:18P MANAGER.Z

6 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    6 EXEC; INCL 5 SESSIONS
    0 SUSP
JOBFENCE= 0; JLIMIT= 3; SLIMIT= 16
```

The status of the job is:

#J35	job number 35
EXEC	job is in execution state
QUIET	the SETMSG OFF command is in effect
5	the logical device number of the standard input device is 5
S	the standard input device is spooled
LP	the standard list device is device class LP (line printer)
THU 5:53P	the job was introduced at 5:53 p.m. on Thursday
MEMBER1.PAYACCT	by user MEMBER1 of the PAYACCT account

For a complete explanation of the various :SHOWJOB formats and options, as well as other job related commands, see the *MPE Commands Reference Manual,* Section VI.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Sections IV and VI)

maintaining a current set of production data files

The technique illustrated in this example can be used for maintaining a set of production data files created by the most recent job runs. For example, a program to update the shipment records may be run each day. Each run creates a file and it is important to save the files created for at least four days. The job shown below will maintain such a set of four files.

```
:JOB     MEMBER1.PAYACCT, GROUP1                  ← Job begins.
PRIORITY=ES.  INPRI=8
JOB NUMBER = #J150
THU, JAN 12, 1978
HP32002B.00.00
:COMMENT ------------------------------------
:COMMENT             DAILY SHIPMENTS UPDATE
:COMMENT ------------------------------------
:PURGE WORKTEMP                                   ← Work file is purged.
FILE NOT FOUND                                    ← If WORKTEMP does not exist, message is printed. New work file is
:BUILD WORKTEMP; DISC=500                           created.
:RUN SHIPUP                                       ← Today's run of SHIPUP uses the work file for output.


 END OF PROGRAM
:PURGE FILE1                                      ← If program runs successfully, oldest file, FILE1, is purged.
:RENAME FILE2,FILE1                               ← FILE2 is renamed FILE1.
:RENAME FILE3,FILE2                                 FILE3 is renamed FILE2.
:RENAME FILE4,FILE3                                 FILE4 is renamed FILE3.
:RENAME WORKTEMP,FILE4                              Work file is renamed FILE4.
:EOJ                                              ← Job terminates.

CPU SEC.=3.  ELAPSED MIN.=1.  THU, JAN 12, 1978, 3:01 PM
```

The line printer listing shown above is printed when the job is executed. The job may be punched in cards and entered through a card reader or stored in a file and initiated with a :STREAM command.

Before the first execution of this job, you must build files FILE2, FILE3, and FILE4 or the :RENAME command will fail. Alternatively, you can insert a :CONTINUE command before each :RENAME command and the job will continue in spite of the :RENAME errors.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Sections IV and VI)

passing data in temporary files

There are two techniques for passing data from one program to another through a temporary disc file. The first one uses the TEMP parameter of the :FILE command.

```
:FILE OUT=X,NEW; REC=-80,,,ASCII; TEMP
:RUN PROG1
```

Define a temporary file called X to be used for the program's OUT file.

```
ENTER DATA
135-22-77771JOHNSON       MYRLE M
```

PROG1 prompts for some data.

```
ENTER DATA
*
```

PROG1 recognizes an asterisk as an indication that there is no more data.

```
 END OF PROGRAM
:FILE IN=X, OLDTEMP
:RUN PROG2
```

Specify the temporary file named X created by PROG1 as the file referenced by IN in PROG2.

```
JOHNSON
```

PROG2 displays the last name from the temporary file's record.

```
 END OF PPOGRAM
```

The OLDTEMP parameter tells MPE that X is an existing temporary file. Since this is so, it will be deleted automatically at the end of the session.

A :BUILD command with the TEMP parameter can also be used to create the temporary file.

The second method for passing data in a temporary file is to use the system's $NEWPASS/$OLDPASS files.

```
:FILE OUT=$NEWPASS; REC=-80,,,ASCII
:RUN PROG1

ENTER DATA
135-22-77771JOHNSON      MYRLE M

ENTER DATA
*
```

Notify the system that the program file named OUT is to be the system temporary file named $NEWPASS.

```
 END OF PROGRAM
:FILE IN=$OLDPASS
:RUN PROG2

JOHNSON

 END OF PROGRAM
```

When PROG1 closes the file named $NEWPASS, the system changes its name to $OLDPASS automatically. Therefore, PROG2's file named IN can be equated to $OLDPASS which contains the desired data.

The file named $OLDPASS exists until the session or job terminates or another file referenced as $NEWPASS is closed. If either of these events occurs, the current $OLDPASS file is deleted.

The :BUILD command cannot be used to create a temporary file named $NEWPASS.

NOTE: The :PREP command and compilers may also use $NEWPASS and $OLDPASS by default, so you must be careful when using them in jobs that also compile or prepare programs.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

creating conditions in a test file using the editor

The Editor can be used to change specific information in a data file. You may find this technique useful when testing an application by creating different conditions in a test file.

```
:EDITOR

HP32201A.5.01 EDIT/3000  TUE, JAN 25, 1977,  1:08 PM
(C) HEWLETT-PACKARD CO. 1976
/T EMPLOYEE,UNN
/CQ 72/72 TO "N" IN 10/15
/K EMPTEST2,UNN
/E
```

:EDITOR — Initiate Editor execution. Note that abbreviated versions of the Editor commands are used in this example.

/T EMPLOYEE,UNN — Move EMPLOYEE file contents into work file.

/CQ 72/72 TO "N" IN 10/15 — Change character position 72 to an "N" in records 10 through 15.

/K EMPTEST2,UNN — Save revised file in a new file named EMPTEST2 and terminate Editor.

The CQ command is an abbreviation of the CHANGEQ command. If you include the Q, the changed lines are not displayed as they are changed.

If you omit the IN 10/15 clause, only one record is changed and it is the first record in the file just moved to the work file. The starting character position and stopping character positions are indicated by 72/72. If you omit the stopping character position (or column) the "N" is *inserted before* column 72.

```
/T EMPLOYEE,UNN
/FIND "SPIEGLE"
    5        275-86-38275SPIEGLE ROSALIE I 22 N. 3RD STREET SJ FSSF35
YY                    (13 )^
/CHANGE 1/11 TO "
    5                   5SPIEGLE ROSALIE I 22 N. 3RD STREET SJ FSSF35
YY
/K EMPTEST1,UNN
/E
```

5 — *line number 5*

YY — *end of line*

(13)^ — *indicates character position 13*

/T EMPLOYEE,UNN — Move EMPLOYEE file contents into work file.

/FIND "SPIEGLE" — Locate record for employee named SPIEGLE.

/CHANGE 1/11 TO " — Change character positions 1 through 11 to blanks.

/K EMPTEST1,UNN — Save revised file in a new file named EMPTEST1 (without line numbers).

To test a program that is supposed to detect a missing social security number, the social security number in a specific record in the file is changed to blanks. This example assumes that the record number for the EMPLOYEE record containing information about Spiegle was not known before the FIND command was used. If the line number is known, the Editor CHANGE or MODIFY command can be used with the line number specified as the *rangelist.*

In the next example, all employee records that contain a salaried employee code (S) and a full-time code (F) are changed to an hourly employee code (H).

```
/T EMPLOYEE,UNN
/SET LEFT=58, SHORT
/FINDQ FIRST
/WHILE
/    FIND "SF"
/    CHANGE 58/58 TO "H"
     5      SF35000 0 750YY
      (58 )^
     5      HF35000 0 750YY
     6      SF35000 4   0NY
      (58 )^
     6      HF35000 4   0NY
     7      SF20000 2   0NY
      (58 )^
     7      HF20000 2   0YY
     8      SF20000 3   0YY
      (58 )^
     8      HF20000 3   0YY
     9      SF15000 5   0YY
      (58 )^
     9      HF15000 5   0YY
    10      SF15000 2   0NY
      (58 )^
    10      HF15000 2   0NY
    11      SF22000 1   0NY
      (58 )^
    11      HF22000 1   0NY
    12      SF22000 0   0YY
      (58 )^
    12      HF22000 0   0YY
    13      SF27000 2   0YY
      (58 )^
```

Move EMPLOYEE data into work file.

Set left margin at character position 58, and request short messages with SHORT parameter. Set pointer to beginning of work file. Enter WHILE command which causes next two commands to be repeated until the condition in the first command can no longer be met.

Character positions 58 through 80 are searched for the string "SF." If the string is located, character position is changed to H.

If you want to suppress the display of these lines, use the FINDQ or FQ and CHANGEQ or CQ commands.

```
    13      HF27000 2    0YY
    14      SF27000 0    0YY
      (58 )^
    14      HF27000 0    0YY
    15      SF45000 01000YN
      (58 )^
    15      HF45000 01000YN
*21*STRING NOT FOUND BEFORE LIMIT
AT DEPTH 2
/SET LEFT=1
/K EMPTEST3,UNN
/E
```

When the last record has been examined by the Editor this message is printed. Set the left margin to the first character position and save the file in a new file named EMPTEST3.

The FINDQ FIRST command is not necessary if the data has just been moved into the work file but it should be used after any Editor commands that move the pointer, for example, LIST or MODIFY.

The SHORT option of the SET command inhibits display of the commands in the WHILE block.

ADDITIONAL INFORMATION:

EDIT/3000 Reference Manual

scanning for conditions set in a disc file by an application program

When you are developing application programs, you may want to examine a test file to see if a particular field contains the correct value. One method for examining the file is to use the FCOPY utility program.

```
:RUN TESTPROG
          .
          .
          .
:RUN FCOPY.PUB.SYS
```

Initiate FCOPY execution after TESTPROG terminates.

```
HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=EMPLOYEE; TO=; SUBSET="Y",71
*200*
```

Copy each EMPLOYEE record that contains a "Y" in character position 71 to the terminal.

FCOPY warns that records are not same size.

```
527-58-31101BOWERS      SUSAN A    13 OAKWOOD DRIVE SC FSHF  400 1   0YY
678-99-33444PARK        RICHARD R  1290 FRIAR WAY   CU MSHF  400 1   0YY
089-23-51023OSGOOD      JOSEPH K   4567 CALIF. ST.  MV MSHF  450 3   0YY
275-86-38275SPIEGLE     ROSALIE I  22 N. 3RD STREET SJ FSSF35000 0 750YY
499-73-82472PIERCE      EMMA R     460 SARATOGA AVE.SCLFMSF20000 2   0YY
253-48-00033PAPADAKIS   JAMES L    1347 KINGSWORTH  CU MMSF20000 3   0YY
537-58-23454MAC LEAN    VINCENT T  819 PALO ALTO ST.MV MMSF15000 5   0YY
212-83-95432FERGUSON    KATHERINE L729 UNION AVENUE LG FMSF22000 0   0YY
456-97-00543GOMEZ       ROBERT D   1331 POETT LANE  SCLMMSF27000 2   0YY
322-11-35874YEE         JANICE K   310 BEEKMAN PLACECU FSSF27000 0   0YY
142-67-55335ALEXANDER   GEOFFREY N 702 CORVALLIS DR.SCRMSSF45000 01000YN
342-48-83295POWELL      PATRICIA J 249 FREMONT AVE. LA FMHF    00 4   0YY
244-75-38445JENSEN      HILDSA A   17650 SHADOW LANELG FMHF 1000 0 800YY
466-39-44334OROZCO      JOSEPH N   112 BRIGHTEN AVE.MV MMHP  800 2   0YY
589-24-21124SCRUGGS     RODNEY X   76 AUBURN DRIVE  CU MMHP  800 1   0YY
300-24-36664JUNG        DIANA T    965 WINCHESTER   SCLFMHP  850 2   0YY
EOF FOUND IN FROMFILE AFTER RECORD 19

16 RECORDS PROCESSED *** 0 ERRORS

>EXIT

 END OF PROGRAM
```

character position 71

When the TO-file is not specified, records are copied to the $STDLIST device which is the terminal in session mode and the line printer in job mode.

In this example, 16 of the 20 records in EMPLOYEE contain a "Y" in character position 71.

You can also use the Editor to list the relative record numbers of records containing particular characters in a specific field.

```
:EDITOR

HP32201A.5.01 EDIT/3000  TUE, JAN 25, 1977,  1:03 PM
(C) HEWLETT-PACKARD CO. 1976
/T EMPLOYEE,UNN
/SET LEFT=58, RIGHT=58, SHORT
/FIND FIRST
    1     H
     (58 )^
/WHILE
/   FINDQ "S"
/   LIST *
    5     S
    6     S
    7     S
    8     S
    9     S
   10     S
   11     S
   12     S
   13     S
   14     S
   15     S
*21*STRING NOT FOUND BEFORE LIMIT
AT DEPTH 2
/E
CLEAR? Y

 END OF SUBSYSTEM
```

← Initiate Editor execution.

← Move contents of EMPLOYEE into work file.

← Set left margin and right margin to character position 58 and request short messages.

← Set pointer to first character position in file (within current margins).

← Use WHILE command to search all records in the file for an "S" in character position 58. List the records that contain an S.

← This message is printed when the end-of-file is encountered.

← Terminate the Editor.

The FIND FIRST command is only necessary if some Editor command has moved the pointer from the first record.

The addition of Q to the FIND command suppresses printing of each record when it is located.

ADDITIONAL INFORMATION:

FCOPY Reference Manual
EDIT/3000 Reference Manual

backup and recovery of files

Making Backup Copies of Disc Files on Magnetic Tape or Serial Disc
Restoring Disc Files from Backup Tape or Serial Disc

making backup copies of disc files on magnetic tape or serial disc

The MPE :STORE command can be used to store one or more files on a backup tape or serial disc. The files are copied in a special format along with all descriptive information (such as account name, group name, and lockword). They can be copied back to the disc with the :RESTORE command as shown in the next example. They can also be transferred to another 3000 system if that system has an account, a user, and a group defined with the same names.

```
:FILE BACKUP1; DEV=TAPE
:STORE @.GROUP1; *BACKUP1; SHOW
LOCKWORD: CUSTOMER.GROUP1.PAYACCT?

LUCKWORD: FILEXYZ.GROUP1.PAYACCT?

  FILES STORED =16

    FILE     .GROUP    .ACCOUNT    LDN   ADDRESS

    DISCARD .GROUP1    .PAYACCT    2     %177372
    DISCF    .GROUP1   .PAYACCT    2     %234542
    DISCFILE.GROUP1    .PAYACCT    2     %161014
    DISCFL   .GROUP1   .PAYACCT    2     %77370
    EMP1     .GROUP1   .PAYACCT    2     %1665
    EMP2     .GROUP1   .PAYACCT    2     %1374
    EMPALL   .GROUP1   .PAYACCT    2     %2614
    EMPFILE .GROUP1    .PAYACCT    2     %364711
    EMPLOYEE.GROUP1    .PAYACCT    2     %17555
    FILEONE .GROUP1    .PAYACCT    2     %76
    FILETWO .GROUP1    .PAYACCT    2     %241760
    FILEXXX .GROUP1    .PAYACCT    2     %253115
    FILEXYZ .GROUP1    .PAYACCT    2     %42426
    HOURLY   .GROUP1   .PAYACCT    2     %205422
    SORTEMPL.GROUP1    .PAYACCT    2     %2567
    XFILE    .GROUP1   .PAYACCT    2     %341

  FILES NOT STORED = 1

    FILE     .GROUP    .ACCOUNT FILESET     REASON

    CUSTOMER.GROUP1    .PAYACCT    1     FILE LOCKWORD WRONG
```

Define tape devicefile named BACKUP1.

Request that all files in GROUP1 be copied to the backup tape and that the results of the store operation be displayed.

You must provide lockwords for files that have them defined.

The logical device number of the disc from which the files are copied is 2.
The address of each file is displayed in octal format.

The CUSTOMER file was not stored since the lockword provided was incorrect.

When the system file opens the TAPE, a message like the following one is displayed on the system console:

```
14:15/#S1123/55/LDEV# FOR "BACKUP1" ON
  TAPE (NUM)?

=REPLY 55,7
```

process identification number

Respond with Control A, REPLY, the process identification number, and the logical unit number of the tape device.

The at sign (@) indicates that you want all the files in the specified group copied. If your log-on group is the group you want to copy, you can omit the group name and period preceding it.

The asterisk preceding the BACKUP1 file name indicates that the file has been defined in a previous :FILE command.

The SHOW parameter requests a summary of the results of the copy operation.

If you are using a CRT instead of a hard copy device, use a :FILE SYSLIST; DEV=LP command to direct the list of files to the line printer. Otherwise you may lose information when the screen is filled.

To copy the same set of files to serial disc, enter:

```
:FILE BACKUP1;DEV=SDISC
:STORE @.GROUP1;*BACKUP1;SHOW
```

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

restoring disc files from backup tape or serial disc

To copy the files to the disc from a back up tape created with the :STORE command, use the :RESTORE command. The :RESTORE command can also be used to copy files from a tape created by the system supervisor using :SYSDUMP.

```
:HELLO MEMBER1.PAYACCT
```

Log on to account to which files are to be restored.

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
```

```
:FILE BACKUP1; DEV=TAPE
:RESTORE *BACKUP1; @; KEEP; SHOW
LOCKWORD: FILEXYZ.GROUP1.PAYACCT?
```

Define tape devicefile named BACKUP1. Copy files from the tape to your log-on group and account (GROUP1 and PAYACCT in this example) if they are not there already.

```
  FILES RESTORED = 4

    FILE     .GROUP    .ACCOUNT   LDN  ADDRESS

    EMPFILE .GROUP1   .PAYACCT   2    %232
    FILEONE .GROUP1   .PAYACCT   2    %200513
    FILETWO .GROUP1   .PAYACCT   2    %200713
    FILEXXX .GROUP1   .PAYACCT   2    %202746
```

Only four files are restored since remaining file names already exist in the group. If KEEP parameter is omitted, file from tape would replace file on disc with same name.

Note that the files are copied to available areas of the disc and are not located in the same areas that they were previously.

```
  FILES NOT RESTORED = 12

    FILE     .GROUP    .ACCOUNT FILESET      REASON

    DISCARD .GROUP1   .PAYACCT     1      ALREADY EXISTS
    DISCF   .GROUP1   .PAYACCT     1      ALREADY EXISTS
    DISCFILE.GROUP1   .PAYACCT     1      ALREADY EXISTS
    DISCFL  .GROUP1   .PAYACCT     1      ALREADY EXISTS
    EMP1    .GROUP1   .PAYACCT     1      ALREADY EXISTS
    EMP2    .GROUP1   .PAYACCT     1      ALREADY EXISTS
    EMPALL  .GROUP1   .PAYACCT     1      ALREADY EXISTS
    EMPLOYEE.GROUP1   .PAYACCT     1      ALREADY EXISTS
    FILEXYZ .GROUP1   .PAYACCT     1      ALREADY EXISTS
    HOURLY  .GROUP1   .PAYACCT     1      ALREADY EXISTS
    SORTEMPL.GROUP1   .PAYACCT     1      ALREADY EXISTS
    XFILE   .GROUP1   .PAYACCT     1      ALREADY EXISTS
```

The tape file name must be preceded by an asterisk to indicate it has been defined with a previous :FILE command. The files are copied to the group from which they came. The group must be either your log-on or home group. In this example they are copied to MEMBER1's home group, GROUP1.

The KEEP parameter specifies that a file is not to be copied if a file with the same name already exists in the group. By using this parameter you can selectively load files without using a separate :RESTORE command for each file.

The DEV= parameter may be used with the :RESTORE command if you want the files to go to a particular device; for example, on a system which has one disc configured with device class name DISC and another with device class name DISCXXX.

The system displays a message on the system console to request the logical device number for the tape file.

```
?IO/10:13/#S95/26/LDEV# FOR "BACKUP1" ON TAPE (NUM)
=REPLY 26,7
```

PIN

Respond with Control A, REPLY, the process identification number (PIN), and the logical unit number.

Note that although the CUSTOMER disc file was not restored, it still exists in the group. The list of files resulting from the SHOW parameter includes only those files which are on the back up tape and not all files in the group.

If you are operating in session mode, you may want to use the :FILE SYSLIST; DEV=LP command to direct the list of restored files to the line printer.

To restore the same set of files from serial disc, enter:

```
:FILE BACKUP1;DEV=SDISC
:RESTORE *BACKUP1;@;KEEP;SHOW
```

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VI)

private volumes

private volumes facility

Users with the Volume Set Usage (UV) capability can maintain files on private disc volumes. These private volumes consist of removable disc packs which, when mounted on a disc drive, can be accessed by MPE through the MPE Private Volumes Facility.

home volume set

A home volume set is the volume set assigned to your group when the group is created with the :NEWGROUP command or altered with the :ALTGROUP command. Files belonging to the group are located on the home volume set; the home volume set is the set referenced by you when you log on under the group. The set need not be mounted until such time as you attempt to access file space in the set. At this time, MPE generates a console request for the Console Operator to mount the set. You can explicitly request that the volume set be mounted before any file access is attempted, and can release the set after you are through with it.

creating a new file on a volume set

Any MPE operation which creates a temporary or permanent file (Editor, :BUILD command, FCOPY, and so forth) will cause the file to be created on a volume set if the home volume set for the group/account is a private volume set.

When creating a file on a home volume set with the :BUILD command, the DEV parameter can be specified in one of the following ways:

$$\text{DEV} = \left\{ \begin{array}{l} \textit{ldn} \\ \textit{devclass} \\ * \\ {}^*\textit{vcname} \\ {}^{**}\textit{volname} \end{array} \right\}$$

The preceding parameters have the following meanings:

ldn	(logical device number) The file will be allocated to that *ldn* only if one of the volumes in the home volume set currently occupies the *ldn.*
devclass	(device class) The file will be allocated within the home volume set's volumes which currently reside on devices of this class.
*	The file will be allocated to any of the volumes of the home volume set. (* is the default.)
**vcname*	(volume class name) If *vcname* is a member of your home volume set, the file is allocated to any of the volumes within the volume class.
***volname*	(volume name) If *volname* is a member of your home volume set, the file is allocated to that volume.

An example of using the :BUILD command to create a new file on a volume set/class is as follows:

```
:BUILD VFILE;DISC=500,10,1;REC=-80;DEV=VCLASS1
```

Name of existing volume class

NOTE

A file may (depending on the DEV parameter specified) reside on more than one volume of a volume set, but must remain within that volume set (may not span more than one volume set). The restrictions are as follows:

DEV=*	The file may reside on more than one volume of the volume set.
DEV=*ldn*	The file is confined to that *ldn*.
DEV=***volname*	The file is confined to the volume whose name is *volname*.
DEV=*devclass*	The file is confined to volumes which reside on devices of that device class.
DEV=**vcname*	The file is confined to volumes within this volume class.

mounting and dismounting volume sets

You can initiate the mounting of a volume set by:

1. Using the :MOUNT command (explicit job/session request).
2. Using a command such as :BUILD which requires that the home volume set be mounted (implicit job/session request).
3. Running a program or subsystem which requires access to a file of a group and account whose volume set is not mounted (programmatic request).

If the volume set is not physically mounted, MPE sends a message to the system console requesting the operator to mount the required volume set.

EXPLICIT JOB/SESSION MOUNT/DISMOUNT REQUEST. You can request a mount of a volume set during a job/session by entering the :MOUNT command. A volume set so assigned for mounting can be released and made available by entering the :DISMOUNT command. This method of mounting/dismounting a volume set is used when only a few consecutive job/session steps require the same volume set.

An example of the :MOUNT command is as follows:

:MOUNT	*Specifies home volume set for log-on group and account. (Default if no parameter specified.)*
:MOUNT VCLASS1	*Specifies volume class name VCLASS1.*

The requesting job/session is suspended until the mount is completed or rejected.

The :DISMOUNT command informs MPE that the specified volume set is no longer needed by the requesting job/session. If there are no other users of the volume set, MPE sends a message notifying the Console Operator that the drives formerly in use are now available to the system. (Once the volume set is dismounted, the device(s) on which it resides are returned to the available state.)

An example of the :DISMOUNT command:

```
:DISMOUNT VCLASS1
```

IMPLICIT JOB/SESSION MOUNT REQUEST. Various user commands which cause access to your log-on group's home volume set will initiate mount requests if the volume set is not mounted already. An example of one of these commands (:BUILD) is as follows:

```
:BUILD VFILE;DISC=500,10,1;REC=-80;DEV=VCLASS1
```

PROGRAMMATIC MOUNT REQUEST. You may issue an FOPEN call referencing a file residing on an unmounted volume set; this causes an implicit user-initiated mount request.

ADDITIONAL INFORMATION:
MPE Commands Reference Manual (Section VI)
MPE Intrinsics Reference Manual (Section III)

MPE tape labels

MPE tape labels

MPE provides a capability which allows you to read and write labels on magnetic tape files. Labeled magnetic tape files can be used to:

1. Identify magnetic tape volumes (reels).
2. Protect tape volumes from inadvertent destruction due to overwriting.
3. Protect private information.
4. Facilitate information interchange between computer systems.

With this MPE tape label capability, you can read, but not write, IBM-standard tape labels and you can read and write ANSI-standard (American Standard Magnetic Tape Labels for Information Interchange x3.27-1969) labels.

Additionally, you can write and read user-defined labels on labeled magnetic tapes.

Due to MPE tape labels, one of the following messages will be displayed on the system console each time a magnetic tape is mounted and recognized:

```
hr:min/#O0/VOL UNLABELED MOUNTED ON LDEV#n
```

(if tape is unlabeled)

```
hr:min/#O0/VOL volid (ANSI) MOUNTED ON LDEV#n
```

(if tape has an ANSI-standard label)

```
hr:min/#O0/VOL volid (IBM) MOUNTED ON LDEV#n
```

(if tape has an IBM-standard label)

opening a labeled magnetic tape file

The following convention must be followed in order to issue a :FILE command for a labeled magnetic tape file:

:FILE *formaldesignator=filename* [/*lockword*]

where

filename	Consists of up to eight alphanumeric characters beginning with a letter that names the file to be processed. The group name may be included.
lockword	Up to eight alphanumeric characters beginning with a letter that protects the file from illegal access. If byte 54 of HDR1 label is a space, there is no security protection for the file. If byte 54 is any other character, then security protection is associated with the file. MPE tape labels will ignore security for IBM-standard labels. Security will also be ignored for ANSI-standard labels unless byte 54 = %230. If a *lockword* is specified on an output file, then byte 54 of HDR1 label will be coded as %230 and the *lockword* will be written in a portion of the HDR2 label. This will enable users to read ANSI-standard and IBM-standard tapes written by other systems, but will provide *lockword* security for ANSI-standard labeled tapes written by MPE.

For example, to open the file whose formal designator is TAPEFILE under the actual designator TEST1, enter:

```
:FILE TAPEFILE=TEST1
```

writing a label on a magnetic tape file

The :FILE command or FOPEN intrinsic is used to write ANSI-standard tape labels (MPE will not write IBM-standard labels).

The LABEL parameter of the :FILE command is used to write a file label on magnetic tape. For example, the statement

```
:FILE TAPEFILE=TEST1;LABEL=FIL001,ANS,12/31/77,NEXT
```

writes a file label as follows:

Volume Identification:	FIL001
Label Type:	ANS (ANSI)
Expiration Date:	12/31/77. This is the date after which the file can be overwritten. If you attempt to overwrite the file before this date, MPE will send a message to the Console Operator asking for confirmation that such is really desired. This affords an extra measure of protection against inadvertently destroying a tape by overwriting when a WRITE RING is left in the tape by mistake.
Sequence:	NEXT. Signifies that the tape is to be positioned at the next file on the tape.

ADDITIONAL INFORMATION:
MPE Commands Reference Manual (Section VI)
MPE Intrinsics Reference Manual (Section III)

reading a label on a magnetic tape file

See the *MPE Intrinsics Reference Manual* for information on reading a label on a magnetic tape file.

listing system information

reporting on the use of account and group resources

There are many ways to get information about system use. The MPE :REPORT command displays accounting information about your log-on group and account. If you are the account manager you receive information about all groups in the account.

```
:HELLO MEMBER1.PAYACCT
```

Initiate session as user MEMBER1 of account PAYACCT. (By default, the log-on group is MEMBER1's home group, GROUP1.)

```
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM

:REPORT
ACCOUNT          FILESPACE-SECTORS        CPU-SECONDS         CONNECT-MINUTES
   /GROUP           COUNT      LIMIT    COUNT      LIMIT     COUNT      LIMIT

PAYACCT               980         **      179         **      1463         **
   /GROUP1            912         **      163         **      1438         **
```

Request information about PAYACCT and GROUP1.

** indicates no limit is defined

CPU time is the total central processor time used

Connect time is the amount of time sessions have used.

If you are the account manager, you receive information about all groups in the account.

```
:HELLO MANAGER.PAYACCT

HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM


:FILE PRINTER; DEV=LP
:REPORT, *PRINTER
```

Account manager logs on to PAYACCT account.

Define PRINTER as a devicefile with device class LP. Specify that the report is to be printed on the PRINTER file instead of $STDLIST (the default). The asterisk indicates the file has been defined with a :FILE command.

If the :REPORT command is used in a job, the report is displayed on the line printer automatically since the line printer is the $STDLIST device in job mode.

The line printer output looks like this:

```
ACCOUNT        FILESPACE-SECTORS       CPU-SECONDS       CONNECT-MINUTES
   /GROUP        COUNT     LIMIT     COUNT     LIMIT     COUNT     LIMIT

PAYACCT           1918        **       300        **      1577        **
   /GROUP1        1765        **       263        **      1510        **
   /GROUP2         153        **        14        **        18        **
   /PUB              0        **        23        **        49        **
```

ADDITIONAL INFORMATION.

MPE Commands Reference Manual (Section V)

listing information about users, groups, and accounts

The LISTDIR2 utility program (LISTDIR on pre-Series II systems) displays information about your log-on account and groups, files and users in the account. You can request any type of information but your user name and account must grant you the appropriate capability or the information will not be supplied. System managers and account managers have access to more information than other users.

The information is displayed on the $STDLIST device unless you use a :FILE command to specify some other device for the LISTDIR2 output file. For example, if you are operating in session mode and want the information to be printed on the line printer, use this :FILE command:

```
:FILE PRINTER; DEV=LP
```

and include a back reference to the file PRINTER when you enter a LISTDIR2 command.

If you are a system manager, system supervisor, or account manager, you can request some of this information with the MPE commands:

- :LISTACCT
- :LISTGROUP
- :LISTUSER
- :LISTF (available to all users).

However, the LISTDIR2 utility program displays the information in a more useful format.

A user who does not have manager or supervisor capability is only permitted to list information about the log-on account, group, and user name.

Now let's look at some examples of LISTDIR2 information, as provided to the account manager of the sample account PAYACCT.

ADDITIONAL INFORMATION:

MPE System Utilities Reference Manual (Section VIII)
System Manager/System Supervisor Reference Manual (Appendix D)

THE LISTDIR2 HELP COMMAND

The first time you use the LISTDIR2 utility program you may want to ask for instructions with the HELP command.

```
:HELLO MANAGER.PAYACCT
HP3000 / MPE III B.00.00.  TUES, MAY 16, 1978, 2:00 PM
:RUN LISTDIR2.PUB.SYS
```

Log on to account in which you are interested. (You must have access to the account).

Initiate execution of LISTDIR2 which is in the PUB group of the SYS account.

```
LISTDIR2 (B.DEC20.1)  (C)HEWLETT-PACKARD CO. 1977
NEW FEATURES: GENERIC NAMES
              MOUNT/DISMOUNT COMMANDS
TYPE 'HELP' FOR AID
>HELP
```

Enter HELP to receive instructions.

```
CONTROL-Y MAY BE TYPED ANYTIME TO STOP THE OUTPUT
FROM ANY COMMAND.

COMMAND SYNTAX IS:
  LISTACCT  [<ASET>] [,<LISTFILE>] [;PASS]
  LISTGROUP [<GSET>] [,<LISTFILE>] [;PASS]
  LISTUSER  [<USET>] [,<LISTFILE>] [;PASS]
  LISTSEC    <FSET>  [,<LISTFILE>] [;PASS]
  LISTF      <FSET>  [,<LISTFILE>] [;PASS] [;MAP]
  HELP      [<LISTFILE>]
  EXIT

  <ASET>      IS AN ACCOUNT NAME OR @.
  <GSET>      IS A GROUP NAME, OPTIONALLY QUALIFIED BY AN
                 ACCOUNT NAME; OR @, OPTIONALLY QUALIFIED BY
                 AN <ASET>.
                 EXAMPLES:  LISTGROUP MYGROUP.MYACCT
                            LISTGROUP @.MYACCT
                            LISTGROUP @.@
  <FSET>      IS A FILE NAME, OPTIONALLY QUALIFIED BY A
                 GROUP AND ACCOUNT NAME; OR @, OPTIONALLY
                 QUALIFIED BY A <GSET>.
```

Brackets indicate the parameter is optional.

```
CONTINUE? (Y/N)Y
```

If you do not want any more instructions, enter an "N" and press return.

```
                EXAMPLES:  LISTF MYFILE.MYGROUP.MYACCT
                           LISTF @.MYGROUP
  <LISTFILE> IS ANY VALID FILE DESIGNATOR, PRECEDED BY
                AN ASTERISK TO BACK-REFERENCE A :FILE
                COMMAND. WITHOUT AN ASTERISK, THE FILE
                DESIGNATOR MUST BE AN EXISTING TEMPORARY
                OR PERMANENT FILE.
                EXAMPLE: LISTACCT @, *PRINTER
   "PASS" PERMITS ACCOUNT AND SYSTEM MANAGERS TO DISPLAY
   PASSWORDS, LOCKWORDS, CREATOR ID'S, AND FILE DISC
   ADDRESSES. THE CREATOR OF A FILE IS ALSO PERMITTED
   TO DISPLAY DISC ADDRESSES AND THE CREATOR ID.

   "MAP" PERMITS THE FILE CREATER AND ACCOUNT AND SYSTEM
   MANAGERS TO DISPLAY THE FILE EXTENT MAP.
>
```

Here is an example of sending the LISTDIR2 information to the line printer:

```
:FILE PRINTER; DEV=LP
:RUN LISTDIR2.PUB.SYS
```

Define a file named PRINTER of device class LP and run LISTDIR2.

```
LISTDIR2(00.00)(C) COPYRIGHT HEWLETT-PACKARD CO 1976
TYPE 'HELP' FOR AID

>LISTACCT PAYACCT,*PRINTER
>EXIT
```

When you use a LISTDIR2 command, specify PRINTER as the list file parameter using the backreference format *PRINTER.

```
 END OF PROGRAM
```

The line printer listing looks like this:

```
********************
ACCOUNT: PAYACCT

DISC SPACE: 2874(S)                PASSWORD: **
CPU TIME: 341(SEC)                 LOC ATTR: %0
CONNECT TIME: 1641(MIN)            SECURITY--READ:    AC
DISC LIMIT: UNLIMITED                        WRITE:   AC
CPU LIMIT: UNLIMITED                         APPEND:  AC
CONNECT LIMIT: UNLIMITED                     LOCK:    AC
MAX PRI: 150                                 EXECUTE: AC
GRP INX PTR: %1070
USR INX PTR: %1067
CAP: AM,AL,GL,ND,SF,IA,BA
```

An explanation of this information is given in the Listing Account Information example which follows.

LISTING ACCOUNT INFORMATION

If no account name is specified with the LISTACCT command, information about the log-on account is listed.

```
>LISTACCT                                                    Enter LISTDIR2 utility LISTACCT command.
********************
ACCOUNT: PAYACCT

DISC SPACE: 980(S)                   PASSWORD: **
CPU TIME: 181(SEC)                   LOC ATTR: %0
CONNECT TIME: 1466(MIN)              SECURITY--READ:    AC
DISC LIMIT: UNLIMITED                          WRITE:   AC
CPU LIMIT: UNLIMITED                           APPEND:  AC
CONNECT LIMIT: UNLIMITED                       LOCK:    AC
MAX PRI: 150                                   EXECUTE: AC
GRP INX PTR: %1070
USR INX PTR: %1067
CAP: AM,AL,GL,ND,SF,IA,BA
```

ACCOUNT:	account name
DISC SPACE:	number of permanent file sectors currently used by the account
CPU TIME:	number of seconds of central processor time used by the account members since the account was created or a :RESETACCT command was last used
CONNECT TIME:	number of minutes account members have been connected to the system in session mode
DISC LIMIT:	maximum number of permanent file sectors the account members are authorized to use
CPU LIMIT:	maximum number of seconds of central processor time the account members are authorized to use
CONNECT LIMIT:	maximum number of minutes of connect time the account members are authorized to use
MAX PRI:	highest subqueue priority an account member can request for a process
GROUP INX PTR: USER INX PTR:	information used by systems engineers
CAP:	capabilities available to account members, see the *System Supervisor/System Manager Reference Manual* for a complete description
PASSWORD:	account password (available only to the system manager)
LOC ATTR:	local attribute assigned to account
SECURITY:	type of security restrictions applied to files in the account (In this example, only account members (AC) are allowed any type of access to the file and each account member is granted the same type of access at the account level.)

LISTING GROUP INFORMATION

List information about the log-on group PUB. (No group name is specified.)

```
>LISTGROUP                                        Enter LISTDIR2 utility LISTGROUP command.
********************
GROUP: PUB.PAYACCT

DISC SPACE: 0(S)                  PASSWORD: **
CPU TIME: 13(SEC)                 SECURITY--READ:     ANY
CONNECT TIME: 22(MIN)                       WRITE:    AL,GU
DISC LIMIT: UNLIMITED                       APPEND:   AL,GU
CPU LIMIT: UNLIMITED                        LOCK:     AL,GU
CONNECT LIMIT: UNLIMITED                    EXECUTE:  ANY
FILE INX PTR:; %1071                        SAVE:     AL,GU
CAP: IA,BA
```

List information about GROUP2.

```
>LISTGROUP GROUP2                                 Enter LISTDIR2 utility LISTGROUP command.
********************
GROUP: GROUP2.PAYACCT

DISC SPACE: 68(S)                  PASSWORD: **
CPU TIME: 4(SEC)                   SECURITY--READ:     GU
CONNECT TIME: 5(MIN)                         WRITE:    GU
DISC LIMIT: UNLIMITED                        APPEND:   GU
CPU LIMIT: UNLIMITED                         LOCK:     GU
CONNECT LIMIT: UNLIMITED                     EXECUTE:  GU
FILE INX PTR: %1101                          SAVE:     GU
CAP: IA,BA
```

GROUP:	group name and account name

DISC SPACE:, CPU TIME:, CONNECT TIME:, DISC LIMIT:, CPU LIMIT:, and CONNECT LIMIT: have the same meaning as they do with account but relate only to the specified group.

FILE INX PTR:	used by systems engineers
CAP:	capabilities available to programs saved in the group
PASSWORD:	the group password (available only to system manager or account manager and displayed only if pass parameter is specified)
SECURITY:	type of security restrictions applied to files in the group (In this example, any system user (ANY) can read the files or execute the programs belonging to the PUB group, but only account librarians (AL) or group users (GU) can access files any other way. Only group users can access the files belonging to GROUP2.)

LISTING USER INFORMATION

To list user information, use the LISTDIR2 utility LISTUSER command.

```
>LISTUSER @
********************
USER: MANAGER.PAYACCT

HOME GROUP: PUB          PASSWORD: **
MAX PRI: 150             LOC ATTR: %0
LOGON CNT: 1
CAP: AM,AL,GL,ND,SF,IA,BA
********************
USER: MEMBER1.PAYACCT

HOME GROUP: GROUP 1      PASSWORD: **
MAX PRI: 150             LOC ATTR: %0
LOGON CNT: 0
CAP: ND,SF,IA,BA
********************
USER: MEMBER2.PAYACCT
```

List information about each account user. (At sign (@) indicates all users.)
If no user name is specified and the at sign is omitted, information about the log-on user name is displayed.

```
HOME GROUP: GROUP1        PASSWORD: **
MAX PRI: 150              LOC ATTR: %0
LOGON CNT: 0
CAP: ND,SF,IA,BA
********************
USER: MEMBERA.PAYACCT

HOME GROUP: GROUP2        PASSWORD: **
MAX PRI: 150              LOC ATTR: %0
LOGON CNT: 0
CAP: ND,SF,IA,BA
********************
USER: MEMBERB.PAYACCT

HOME GROUP: GROUP2        PASSWORD: **
MAX PRI: 150              LOC ATTR: %0
LOGON CNT: 0
CAP: ND.SF.IA.BA
```

USER:	user name and account name
HOME GROUP:	user's home group
MAX PRI:	highest subqueue user can request for a process
LOGON CNT:	number of users currently logged on to the system with this user name
CAP:	capabilities granted to the user
PASSWORD:	user's password (available only to account manager or system manager and available only if PASS parameter is specified)
LOC ATTR:	local attribute assigned to user

```
>LISTF EMPLOYEE.GROUP1
*********************
FILE: EMPLOYEE.GROUP1.PAYACCT

FCODE: 0                    FOPTIONS: ASCII,FIXED
BLK FACTOR: 3               CREATOR: **
REC SIZE: 80(B)             LOCKWORD: **
BLK SIZE: 120(W)            SECURITY--READ:    ANY
EXT SIZE: 15(S)                       WRITE:   ANY
# REC: 20                             APPEND:  ANY
# SEC: 15                             LOCK:    ANY
# EXT: 1                              EXECUTE: ANY
MAX REC: 40                         **SECURITY IS ON
MAX EXT: 1                  COLD LOAD ID: %12744
# LABELS: 0                 CREATED: FRI, 21 JAN 1977
MAX LABELS: 0               MODIFIED: FRI, 21 JAN 1977
DISC DEV #: 2               ACCESSED: FRI, 21 JAN 1977
DISC TYPE: 0                LABEL ADR: **
DISC SUBTYPE: 3             SEC OFFSET: %1
CLASS :DISC                 FLAGS: NO ACCESSORS
FCB VECTOR: %0
```

List information about the file named EMPLOYEE that belongs to GROUP1.

FILE:	file name, group name, and account name
FCODE:	file code
BLK FACTOR:	number of logical records per block
REC SIZE.	record size in bytes or characters
BLK SIZE:	block size in words
EXT SIZE:	extent size in sectors
# REC:	number of logical records in use
# SEC:	number of disc sectors in use
# EXT:	number of extents currently allocated
MAX REC:	maximum number of records allowed in file
MAX EXT:	maximum number of extents allowed for file
# LABELS:	number of user labels created
MAX LABELS:	maximum number of user labels allowed
DISC DEV #:	logical device number of the disc device
DISC TYPE:	device type as defined during system configuration
DISC SUBTYPE:	device subtype as defined during system configuration
CLASS:	device class name
FCB VECTOR:	used by systems engineers
FOPTIONS:	file characteristics as defined in FOPEN intrinsic *foptions* parameter.
CREATOR:	user name of file creator (available only to file creator or account or system manager and displayed only if PASS parameter is specified)

LOCKWORD:	file's lockword (available only to account or system manager and displayed only if PASS parameter is specified)
SECURITY:	security provisions defined for the file (In the example ANY indicates that any user can have any type of access that is also allowed at the account and group levels.) **SECURITY is $\begin{matrix}\text{ON}\\\text{OFF}\end{matrix}$ indicates whether file security has been released or secured.
COLD LOAD ID:	number identifying the current cold load when this file was last closed

The *System Manager/System Supervisor Reference Manual* describes file information in more detail. If the file is a program file, additional information is displayed.

CREATED:	date when file was created
MODIFIED.	date when file was last modified
ACCESSED:	date when file was last accessed
LABEL ADR:	disc address of disc file label (available only to creator or account or system manager)
SEC OFFSET:	when added to label address, address of first logical record of file
FLAGS:	specifies why file is locked or if it is not locked specifies NO ACCESSORS (Check this if you cannot gain access to a file.)

LISTING FILE SECURITY INFORMATION

```
>LISTSEC FILEXYZ.GROUP1
********************
FILE: FILEXYZ.GROUP1.PAYACCT

SECURITY--READ:    AC
 (ACCT)   WRITE:   AC
          APPEND:  AC
          LOCK:    AC
          EXECUTE: AC

SECURITY--READ:    GU
(GROUP)   WRITE:   GU
          APPEND:  GU
          LOCK:    GU
          EXECUTE: GU
          SAVE:    GU

SECURITY--READ:    ANY                   FCODE: 0
 (FILE)   WRITE:   ANY                   CREATOR: **
          APPEND:  ANY                   LOCKWORD: **
          LOCK:    ANY                 **SECURITY IS ON
          EXECUTE: ANY

FOR MANAGER.PAYACCT:READ,WRITE,APPEND,LOCK,EXECUTE
```

List file security information for file named FILEXYZ in GROUP1.

FILE:	file name, group name, and account name
SECURITY: (ACCT)	account level security provisions for file (In this example, account members (AC) can access the file in any way.)
SECURITY: (GROUP)	group level security provisions for file (In this example, group users (GU) can access the file in any way.)
SECURITY: (FILE)	file level security provisions for file (In this example, any user who passes the group and account levels of security can access the file in any way.)
FCODE:	file code
CREATOR:	file creator's user name (available only to the creator or system or account manager and displayed only if PASS parameter is specified)
LOCKWORD:	file's lockword (available only to the system or account manager and displayed only if PASS parameter is specified)
**SECURITY IS ON / OFF	Tells whether file security has been released or secured.
FOR MANAGER.PAYACCT:	actual access permitted the log-on user. The account manager overrides this security.

listing devicefile information

If you want to know the status of devicefiles you can use the :SHOWIN and :SHOWOUT commands. The :SHOWIN command displays information about all input devicefiles existing in the system, or specific information about all or anv of them.

```
:SHOWIN

DEV/CL    DFID     JOBNUM   FNAME    STATE  FRM SPACE RANK PRI #C

29        #I238    #S210    $STDIN   OPENED
5         #I243             CARDS    READY        8
                   MEMBER1.PAYACCT


2 FILES:
    0 ACTIVE
    1 READY; INCL 1 SPOOFLES, 0 DEFERRED
    1 OPENED; INCL 0 SPOOFLES
    0 LOCKED; INCL 0 SPOOFLES
    1 SPOOFLES: 8 SECTORS
```

Enter command without any parameters. Information about your session's files is displayed.

Status of $STDIN file is displayed.

Spooled devicefile is ready. This file consists of a :DATA command and data entered through the card reader.

CARDS file is ready. The spooled file consists of 8 disc sectors.

The first line of information tells the status of the $STDIN file which is the standard input device for session number #S210 (MEMBER1's current session in which the :SHOWIN command has been entered.) The logical device number of the terminal on which this session is running is 29 and the device file id number is #I238.

The second line of information tells the status of a spooled file named CARDS that has been read into logical device number 5 (the card reader). It has been assigned device file is #I243 and is in a READY state. This file is using 8 sectors of disc space.

The summary of file information indicates that 2 input devicefiles currently exist for your session; one is a spooled file (spoofle) in the READY state and one is a regular file that is open.

The :SHOWIN command parameters allow you to be selective in the information you display. These are described in the *MPE Commands Reference Manual* with several examples of their use.

Here is an example of the :SHOWOUT command.

```
:FILE PRINTER;DEV=LP
:RUN LISTDIR2.PUB.SYS
```

PRINTER is a devicefile with device class LP.
The LISTDIR2 utility is executed.

```
LISTDIR2(00.00)(C) COPYRIGHT HEWLETT-PACKARD CO 1976
TYPE 'HELP' FOR AID

>LISTF @,*PRINTER
>EXIT
```

A list of information about all files is to be displayed on the line printer.

```
 END OF PROGRAM
:SHOWOUT STATUS
21 FILES:
     1 ACTIVE
     1 READY; INCL 1 SPOOFLES, 0 DEFERRED
    19 OPENED; INCL 1 SPOOFLES
     0 LOCKED; INCL 0 SPOOFLES
     3 SPOOFLES: 656 SECTORS

OUTFENCE=1
```

The :SHOWOUT command lists the status of all current output devicefiles if the STATUS parameter is specified.

1 devicefile is active and 1 is in a READY state. It is a spooled file. 19 files are open including 1 spooled file. A total of 3 spooled files exist and consist of 656 disc sectors.

```
:SHOWOUT SP
DEV/CL    DFID     JOBNUM   FNAME     STATE FRM SPACE RANK PRI #C

6         #0603    #S323    PRINTER   ACTIVE      156    1 13   1

OUTFENCE=1
```

The :SHOWOUT command with the SP parameter displays information about currently spooled files.

The spooled file named PRINTER is in an ACTIVE state (being printed). It is logical device number 6 (the line printer) and is assigned device file id #0603. The file belongs to session number #S323. The file is 156 disc sectors long and has output priority 13 (by default). Rank indicates the order in which the file was entered in the system with respect to other files. #C indicates the number of copies requested, in this case, only 1.

ADDITIONAL INFORMATION:

MPE Commands Reference Manual (Section VIII)

KSAM files

creating a KSAM file

A KSAM file must be created with the KSAMUTIL program's BUILD command, or with the FOPEN intrinsic. Do not use the MPE :BUILD command. Here is an example of using KSAMUTIL.

```
:RUN KSAMUTIL.PUB.SYS
```

Initiate execution of KSAMUTIL.

```
HP32208A.00.00 FRI, FEB  4, 1977,  2:01 PM
>BUILD KFILE;REC=-72,1,F,ASCII&
>      ;KEYFILE=KFILEKEY&
>         ;KEY=B,13,12,,DUPLICATE&
>         ;KEY=B,12,1,,DUPLICATE&
>      ;FIRSTREC=1
```

KSAMUTIL prompts for a command with the greater than (>) symbol. Create a file named KFILE with a key file named KFILE-KEY. The ampersand (&) is used to continue the command on the next line.

```
>VERIFY
```

You can use the VERIFY command to get information about the file.

```
WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?1

KFILE.GROUP1.PAYACCCT       CREATOR=MEMBER1
FOPTIONS(004005)=KSAM, :FILE, NOCCTL, F, FILENAME, ASCII, PERM
AOPTIONS(010404)=AS IS, NOBUF, DEFAULT, NO FLOCK, NO MR, INOUT
RECSIZE:SUB:TYP:LDNUM:DRT:UN.:  CODE:LOGICAL PTR: END OF FILE:FILE LIMIT
    -72:  3:  0:    2:  5:  0:     0:           1:          0:       1023
 LOG. COUNT:PHYS. COUNT:BLK SZ:EXT SZ:NR EXT: LABELS:LDN:   DISCADDR:
          0:           0:   -72:   129:     8:       1:  2:00000112326:

WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?2

KEY FILE=KFILEKEY KEY FILE DEVICE=2         SIZE=        274  KEYS=      2
FLAGWORD(000002)=RANDOM PRIMARY, FIRST RECORD=1, PERMANENT
KEY TY LENGTH   LOC. D KEY BF  LEVEL
  1  B     12     13 Y    100      0
  2  B      1     12 Y    202      0

WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?5

>EXIT

 END OF PROGRAM
```

Enter a 5 to terminate VERIFY command and then terminate KSAMUTIL with an EXIT command.

In the example the following data file characteristics are specified:

- record size of 72 characters
- blocking factor of 1 logical record per block
- fixed-length records
- ASCII data format.
- the first logical record in the file is to be referenced as record 1 (;FIRSTREC=1)

Default values are accepted for all other file characteristics.

In addition to the data file name, a key file name and at least one key field must be specified. In the example, a primary key (the first key entered) and an alternate key are specified.

- The primary key begins in character position 13 and is 12 characters long.
- The alternate key begins in character position 12 and consists of 1 character.

Both keys are type B (character or byte) and both keys allow duplicate values of the key to be entered in different records of the data file.

The VERIFY command provides the following information when you select option 1 (FILE INFO):

- file name (data file name, group name, account name) and creator's user name
- FOPTIONS: the file characteristics specified in the FOPEN intrinsic when the file is opened. See the *MPE Intrinsic Reference Manual* for a description of the *foptions.*
- AOPTIONS: the file access options specified in the FOPEN intrinsic when the file is opened. See the *MPE Intrinsics Reference Manual* for a description of the *aoptions.*
- RECSIZE: number of characters (negative) or words (positive) in record
- SUB: device subtype
- TYP: device type
- LDNUM: logical device number
- DRT: Device Reference Table number
- UN.: the hardware unit number of device
- CODE: the file code
- LOGICAL PTR: the number of the logical record to which the pointer is currently pointing
- END OF FILE: location of end of file (currently record 0). Note that MPE references the first record as record 0 and KSAM references the first record as record 1 as requested in this example with the FIRSTREC=1 parameter.
- FILE LIMIT: number of logical records that can be included in the data file.
- LOG. COUNT: number of logical records passed to and from user during the current access to the file.
- PHYS. COUNT: number of physical input/output operations performed within this process since the file was opened.
- BLK SZ: block size in words (positive) or characters (negative)
- EXT SZ: the size of each extent in sectors
- NR EXT: the number of extents to be allocated for the file
- LABELS: the number of user labels defined for the file
- LDN: logical device number
- DISCADDR: address on disc where first record in file is stored

If you select option 2 (KSAM PARAMETERS) of the VERIFY command, information is printed about each key. The primary key is listed first. The information includes:

- the key number
- the key type (in this example both keys are type B)
- the length of the key in characters
- the location, or character position, of the first character in the key
- a flag to indicate whether or not duplicate values are allowed (Y indicates yes, N indicates no)
- maximum number of keys per block

ADDITIONAL INFORMATION:

KSAM/3000 Reference Manual

copying an MPE file to a KSAM file

You can use the FCOPY utility program to copy an MPE file to a KSAM file but the KSAM file must first be created with the KSAMUTIL BUILD command.

```
:RUN FCOPY.PUB SYS

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

>FROM=EMPLOYEE;TO=KFILE

EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS
```

Initiate FCOPY execution.

Copy contents of MPE file named EMPLOYEE to the KSAM file created in the last example named KFILE. Note that to reference the data and key file, only the data file name need be used.

FCOPY copies the data from the EMPLOYEE file to the KFILE and at the same time generates and writes the necessary key information to the key file named KFILEKEY.

ADDITIONAL INFORMATION:

KSAM/3000 Reference Manual
FCOPY Reference Manual

copying a KSAM file to the terminal

If you copy a KSAM file to another file and do not specify otherwise, the file is copied in ascending order according to the value of the primary key data field.

```
:RUN FCOPY.PUB.SYS                                  Initiate FCOPY execution.

HP32212A.1.03 FILE COPIER (C) HEWLETT-PACKARD CO. 1976

                                                    Copy from KFILE to terminal (omit TO-file name)
>FROM=KFILE;TO=
142-67-55335ALEXANDER   GEOFFREY N 702 CORVALLIS DR.SCRMSSF45000 01000YN
527-58-31101BOWERS      SUSAN A    13 OAKWOOD DRIVE SC FSHF   400 1    0YY
212-83-95432FERGUSON    KATHERINE L729 UNION AVENUE LG FMSF22000 0    0YY
456-97-00543GOMEZ       ROBERT D   1331 POETT LANE  SCLMMSF27000 2    0YY
531-66-77771HORN        MICHAEL S  2230 BRONSON AVE.SJ MMSF22000 1    0NY
432-11-92012HOUGHTON    MARGE S    313 MOONBEAM WAY LA FSHF   450 1    0NY
244-75-38445JENSEN      HILDA A    17650 SHADOW LANELG FMHF 1000 0  800YY
300-24-36664JUNG        DIANA T    965 WINCHESTER   SCLFMHP   850 2    0YY
333-44-82825KONIGSBERG  JOANNE M   45 ELDEN STREET  LA FMSF15000 2    0NY
537-58-23454MAC LEAN    VINCENT T  819 PALO ALTO ST.MV MMSF15000 5    0YY
466-39-44334OROZCO      JOSEPH N   112 BRIGHTEN AVE.MV MMHP   800 2    0YY
089-23-51023OSGOOD      JOSEPH K   4567 CALIF. ST.  MV MSHF   450 3    0YY
253-48-00033PAPADAKIS   JAMES L    1347 KINGSWORTH  CU MMSF20000 3    0YY
678-99-33444PARK        RICHARD R  1290 FRIAR WAY   CU MSHF   400 1    0YY
499-73-82472PIERCE      EMMA R     460 SARATOGA AVE.SCLFMSF20000 2    0YY
342-48-83295POWELL      PATRICIA J 249 FREMONT AVE. LA FMHF 1000 4    0YY
138-66-09021RICHARDS    NORMAN A   1961 MULBERRY DR.LG MMSF35000 4    0NY
589-24-21124SCRUGGS     RODNEY X   76 AUBURN DRIVE  CU MMHP   800 1    0YY
275-86-38275SPIEGLE     ROSALIE I  22 N. 3RD STREET SJ FSSF35000 0  750YY
322-11-35874YEE         JANICE K   310 BEEKMAN PLACECU FSSF27000 0    0YY
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS
                         primary key is last name
```

The contents of the KFILE are displayed on the terminal in ascending order of employees' last names. The primary key is the default key for sequential processing.

If you include the KEY= parameter in the FROM= command, the records are copied according to the ascending order of the key that begins in the specified character position. In the next example, the records are copied according to the values of the alternate key that begins in character position 12.

```
:RUN FCOPY.PUB.SYS
>FROM=KFILE;TO=;KEY=12
```

Copy from KFILE to terminal according to the key beginning in character position 12.

character position 12

```
527-58-31101BOWERS      SUSAN A    13 OAKWOOD DRIVE SC FSHF  400 1   0YY
138-66-09021RICHARDS    NORMAN A   1961 MULBERRY DR.LG MMSF35000 4   0NY
531-66-77771HORN        MICHAEL S  2230 BRONSON AVE.SJ MMSF22000 1   0NY
432-11-92012HOUGHTON    MARGE S    313 MOONBEAM WAY LA FSHF  450 1   0NY
499-73-82472PIERCE      EMMA R     460 SARATOGA AVE.SCLFMSF20000 2   0YY
212-83-95432FERGUSON    KATHERINE L729 UNION AVENUE LG FMSF22000 0   0YY
089-23-51023OSGOOD      JOSEPH K   4567 CALIF. ST.  MV MSHF  450 3   0YY
253-48-00033PAPADAKIS   JAMES L    1347 KINGSWORTH  CU MMSF20000 3   0YY
456-97-00543GOMEZ       ROBERT D   1331 POETT LANE  SCLMMSF27000 2   0YY
678-99-33444PARK        RICHARD R  1290 FRIAR WAY   CU MSHF  400 1   0YY
537-58-23454MAC LEAN    VINCENT T  819 PALO ALTO ST.MV MMSF15000 5   0YY
322-11-35874YEE         JANICE K   310 BEEKMAN PLACECU FSSF27000 0   0YY
466-39-44334OROZCO      JOSEPH N   112 BRIGHTEN AVE.MV MMHP  800 2   0YY
589-24-21124SCRUGGS     RODNEY X   76 AUBURN DRIVE  CU MMHP  800 1   0YY
300-24-36664JUNG        DIANA T    965 WINCHESTER   SCLFMHP  850 2   0YY
275-86-38275SPIEGLE     ROSALIE I  22 N. 3RD STREET SJ FSSF35000 0 750YY
333-44-82825KONIGSBERG  JOANNE M   45 ELDEN STREET  LA FMSF15000 2   0NY
142-67-55335ALEXANDER   GEOFFREY N 702 CORVALLIS DR.SCRMSSF45000 01000YN
342-48-83295POWELL      PATRICIA J 249 FREMONT AVE. LA FMHF 1000 4   0YY
244-75-38445JENSEN      HILDA A    17650 SHADOW LANELG FMHF 1000 0 800YY
EOF FOUND IN FROMFILE AFTER RECORD 19

20 RECORDS PROCESSED *** 0 ERRORS
```

You can also copy a subset of the file by specifying a particular key and a value for the key. In the next example, only records with the character 4 beginning in position 12 are copied.

```
>FROM=KFILE;TO=;SUBSET="4",12
```

Copy from KFILE to terminal according to alternate key.

```
300-24-36664JUNG           DIANA T     965 WINCHESTER    SCLFMHP  850 2    0YY
537-58-23454MAC LEAN       VINCENT T   819 PALO ALTO ST.MV MMSF15000 5    0YY
466-39-44334ORDZCO         JOSEPH N    112 BRIGHTEN AVE.MV MMHP   800 2    0YY
678-99-33444PARK           RICHARD R   1290 FRIAR WAY    CU MSHF  400 1    0YY
589-24-21124SCRUGGS        RODNEY X    76 AUBURN DRIVE   CU MMHP  800 1    0YY
322-11-35874YEE            JANICE K    310 BEEKMAN PLACECU FSSF27000 0    0YY
EOF FOUND IN FROMFILE AFTER RECORD 19

6 RECORDS PROCESSED *** 0 ERRORS
```

character position 12

ADDITIONAL INFORMATION:

FCOPY Reference Manual
KSAM/3000 Reference Manual

changing the name of a KSAM file

You must use the KSAMUTIL program to change the name of a KSAM file. If both the data file and the key file names are to be changed, you must use the KSAMUTIL RENAME command twice.

```
:RUN KSAMUTIL.PUB.SYS                          ← Initiate KSAMUTIL execution.

HP32208A.00.00 FRI, FEB  4, 1977,   2:28 PM
>RENAME KFILE,KEMPLOY                          ← Change the name of KFILE to KEMPLOY.
>RENAME KFILEKEY,KEMPKEY                       ← Change the name of KFILEKEY, to KEMPKEY.
>VERIFY KEMPLOY                                ← Display file information.

WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?1

KEMPLOY.GROUP1.PAYACCT      CREATOR=MEMBER1
FOPTIONS(004005)=KSAM, :FILE, NOCCTL, F, FILENAME, ASCII, PERM
AOPTIONS(010404)=AS IS, NOBUF, DEFAULT, NO FLOCK, NO MR, INOUT
RECSIZE:SUB:TYP:LDNUM:DRT:UN.:  CODE:LOGICAL PTR: END OF FILE:FILE LIMIT
     -72:  3:  0:    2:  5:  0:     0:            1:            20:       1023
 LOG. COUNT:PHYS. COUNT:BLK SZ:EXT SZ:NR EXT: LABELS:LDN:   DISCADDR:
          1:            1:   -72:   129:      8:       1:  2:00000112326:

WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?2

KEY FILE=KEMPKEY  KEY FILE DEVICE=2          SIZE=         274   KEYS=      2
FLAGWORD(000002)=RANDOM PRIMARY, FIRST RECORD=1, PERMANENT
KEY TY LENGTH   LOC. D KEY BF   RESVD
  1  B     12      13 Y    100       0
  2  B      1      12 Y    202       0

WHICH (1=FILE INFO, 2=KSAM PARAMETERS,  3=KSAM CONTROL, 4=ALL, 5=NONE)?5
```

As you see by using the VERIFY command, the KEMPLOY file is the same file and has the same characteristics as KFILE. Its key file is KEMPKEY rather than KFILEKEY. Do not use the MPE :RENAME command.

ADDITIONAL INFORMATION:

KSAM/3000 Reference Manual

deleting a KSAM file

To delete a KSAM file, use the KSAMUTIL program PURGE command.

```
:RUN KSAMUTIL.PUB.SYS

HP32208A.00.00 FRI, FEB  4, 1977,   2:28 PM

>PURGE KEMPLOY
KEMPLOY.GROUP1.PAYACCT & KEMPKEY PURGED.
>EXIT

 END OF PROGRAM
```

← Initiate KSAMUTIL execution.

← Delete the KEMPLOY file.

Note that the key file KEMPKEY associated with the KEMPLOY data file is also deleted.

You can also purge a temporary KSAM file by including the parameter ,TEMP.

The MPE :PURGE command may be used but you must enter it twice, once for the data file and once for the key file.

ADDITIONAL INFORMATION:

KSAM/3000 Reference Manual

glossary

glossary

account	The basic unit to which resources are assigned and the major billing entity to which system resources such as central processor time, on-line connect time and file space are allocated and charged.
account level security	The file access modes and types of users to whom they are available as specified by the system manager when the account is created.
account librarian	A user who is granted special file-access modes by the account manager for maintenance of files within the account. There may be more than one account librarian per account.
account manager	A user who manages the account by defining valid users and file groups and specifying resource-use limits (if any) for them.
account member	Anyone for whom the account manager defines a user name and attributes.
actual file designator	The name by which MPE recognizes a devicefile or a disc file.
allocating	The system process of locating and reserving disc space to be used for a particular file.
alphanumeric character	A character consisting of one of the alphabetic characters (A–Z) or the numeric digits (0–9).
ASCII data	Data formatted in 7-bit characters conforming to the American Standard Code for Information Interchange. See table in the *HP 3000 Software Pocket Guide.*
ASCII files	Files whose records are padded with blanks.
back reference	The technique of using an asterisk before a file name to indicate that the file has been defined in a preceding :FILE command entered during the current session or job.
back up	Typically refers to copies of disc files residing on magnetic tape files.
batch mode or processing	A technique of submitting to the system, as a single unit, commands requesting various MPE operations such as program compilation and execution, file manipulation, or utility functions. Each unit is called a job.
BCD	Binary Coded Decimal. A decimal notation in which individual decimal digits are each represented by a group of binary digits.
binary data	Data consisting of binary numbers. All available data types in the HP 3000 system can be coded as binary data.
bit	Acronym for binary digit. One binary digit (0 or 1).

block	A group of one or more logical records transmitted to or from a file by an input/output operation.
blocking factor	The number of logical records in a block.
buffer	A storage device used to compensate for a difference in rate of flow of data, or time of occurrence of events, when transmitting data from one device to another. An area in which information is temporarily stored during a transfer of information.
byte	A sequence of 8 bits operated upon as a unit. One-half an HP 3000 word.
carriage control information	Directions about the formatting of data on a line printer or terminal, specifically the way in which line feeds and carriage returns are to be handled.
CCTL	A parameter of the :BUILD and :FILE commands that tells MPE that the program will provide carriage control information.
closing a file	Terminating access to a file after completing all input/output operations with the file. Accomplished by calling the FCLOSE intrinsic or terminating program execution.
cold load	The process of loading the MPE system from the system disc to memory. This is the standard operating procedure for starting a system after it has been shut down.
collating sequence, ASCII	The sequence into which the ASCII characters are sorted or ordered (based on the numeric value of the 7 bits that represent the character).
command	A key word that directs MPE, a subsystem, or a utility program to perform a specific operation.
COMP-3	A term used in the COBOL programming language to refer to the packed decimal data format.
connect time	The amount of time between the beginning and ending of a session.
Control A (A^c)	A character entered by pressing the control key and holding it down while pressing the A key.
control key	A special key on terminals used in combination with other keys to create a special character.
Control Y (Y^c)	A character entered by pressing the control key and holding it down while pressing the Y key.
CPU	Central processing unit of the computer, its arithmetic unit and registers. Sometimes called the main frame.
CPU time	The amount of time a user, group, or account has used the central processing unit.

creator, file	User who creates file with the :BUILD command or FOPEN intrinsic (identified by user name, account, and log-on or home group).
CTRANSLATE	An MPE intrinsic that converts a string of characters from EBCDIC to ASCII and vice versa.
data base	A collection of logically-related files containing both data and structural information.
:DATA command	An MPE command that defines a devicefile for the input of information coded on cards.
defaults	Values for parameters or various system characteristics that take effect if a parameter is omitted or a characteristic is not specified.
device class name	A name consisting of up to 8 alphanumeric characters, beginning with a letter, defined by the system supervisor and assigned to one or more devices of a particular type when the system is configured.
device subtype	A number ranging from 0 to 15 defining a specific device and software driver. (See next entry.)
device type	A number defining a particular type of device, for example, 0=moving head disc. See the *System Manager/System Supervisor Reference Manual* for a list of the various device types and subtypes.
device-independent	The characteristic of files referenced by programs to be run on the HP 3000 that allows the type of file or device to be specified when the program is actually run. For example, a program file may be defined as a disc file for one execution of the program and a tape file for another. The MPE file system determines the actual location of the file for you when you run the program.
devicefile	A file originating from or directed to a non-disc device. A file currently being input to or output from any peripheral device except a disc.
direct access	Reading from or writing to a random access device (usually a disc) by addressing a specific logical record.
EBCDIC	Extended Binary Coded Decimal Interchange Code. An 8-bit code representing an extension of BCD. Can represent up to 256 characters.
Editor	The HP 3000 text editor, EDIT/3000, used to create and manipulate ASCII files.
escape key	A special key on terminals, usually identified by the letters ESC, that changes the terminal into a different mode. When followed by a semi-colon (;) it disables the MPE Echo Facility, preventing the characters you enter from being displayed. When followed by a colon (:) it enables the Echo Facility again.
execution state	A term referring to a program's mode when it is actually running on the system.

extent	A portion of a file consisting of contiguous sectors on the disc.
FCLOSE intrinsic	An MPE system-supplied intrinsic that is used to close a file.
FCOPY	The HP 3000 file copier, one of the utility programs supplied with your system.
FID	File Information Display (see below).
file	A collection of related records treated as a unit. An HP 3000 file may be an MPE disc or devicefile, a KSAM disc file, or an IMAGE data set.
file code	A four-digit integer that identifies the special function of a file. You may assign a file code between 0 and 1023 to a file you create to classify it according to its purpose. The system uses number 1024 through 1070 for special system files. (See the *filecode* parameter of the :FILE command in the *MPE Commands Reference Manual* for a list of these file codes.)
file creator	See creator.
file directory	A catalog maintained by the system to record the names, creators, locations, and other characteristics of files known to the system.
file information display	A display of file characteristics, an error message, an error number, and current FOPEN intrinsic parameters that is provided when certain file input-output errors occur. (See the *Error Messages and Recovery Manual* for a description.)
file label	The first sector of a disc file. Contains information about the file such as the information displayed when you enter a :LISTF *name,*2 command. (See the *MPE Commands Reference Manual,* table 6-13 for a list of the label contents.)
file level security	The file access modes and types of users to whom they are available. These are determined by default when the file is created and may be changed by the file creator using the :ALTSEC command which is described in the *MPE Commands Reference Manual.*
file pointer	Logical record pointer kept by MPE to indicate the next sequential record to be accessed in a file. Pointer is set to the first record when the file is opened.
File System	The part of the MPE operating system that automatically handles user access to input/output devices and data blocking, buffering, data transfers, and deblocking.
fixed-length record	A record that always contains the same number of characters or words.
FOPEN intrinsic	An MPE system-supplied intrinsic that is used to open a file.

foptions parameter	A parameter of the FOPEN intrinsic that is used to specify different file characteristics. (See the *MPE Intrinsics Reference Manual* for a description of these characteristics.)
formal file designator	The name by which a program recognizes a file.
FTN*nn*	The formal file designator for a FORTRAN program's file.
fully-qualified file name	A file name qualified by the name of the group and account to which the file belongs, for example, FILEX.GROUPX.ACCOUNTX.
group	A bookkeeping facility that partitions some of an account's file domain and resources into a private sub-domain. The on-line connect-time, central processor time, and disc-file space that you use are charged to your log-on group as well as account.
group level security	The file access modes and types of users to whom they are available as specified by the account manager when the group is created.
group user	A user who is logged on to a particular group, or who is assigned the group as a home group.
hard-wired terminal	A terminal directly connected to the computer system.
home group	The group assigned to a user by the account manager when he defines the user name. This group is the user's default log-on group if a group name is not specified with the :HELLO or :JOB command.
IMAGE	A data base management system available on the HP 3000 consisting of a set of programs and procedures that can be used to define, create, access and maintain a data base.
intrinsic	An operating system procedure used to access and alter files, as well as to perform other operations.
job	A single unit of commands that request various MPE operations such as compiling and executing programs, manipulating files, or performing utility functions, submitted by the user to the HP 3000.
job control commands	MPE, subsystem, and utility commands submitted as part of a job. A set of these commands must begin with a :JOB command and end with an :EOJ command.
key	A user-defined field within a data record. The value of the key field can be used to locate and access the particular data record.
key word	A word that has been assigned a specific meaning by the system.

KSAM	Keyed Sequential Access Method, a file access method for the HP 3000 Series II computer system in which records may be accessed sequentially or randomly by primary or alternate key value.
LISTDIR2	An MPE utility program that lists information about accounts, groups, users and files.
LISTEQ2	An MPE utility program that lists the MPE :FILE commands used, and temporary files created, during the current session or job.
local attribute	A doubleword number defined by system or account managers and used by accounts or groups for any purpose desired. A local attribute may be accessed in a program through the WHO intrinsic.
lockword	A word assigned to a file when it is created that must be supplied to access the file in any way. The word may be from one to eight alphanumeric characters long and must begin with an alphabetic character.
log on	The process of initiating a session with the MPE operating system by using the :HELLO command or initiating a job with the :JOB command.
log-on group	The group you select at log-on time for your local file domain. You must specify a group with the :HELLO or :JOB command, unless you have a home group in which case it is the default log-on group.
logical device number	A unique number in the range of 1 to 255 that the system supervisor assigns to a device when the system is configured.
logical record	A collection of fields or related data treated as a unit, residing in a file.
MERGE	See SORT.
modem	A modulator-demodulator device that provides the appropriate interface between a communications link (such as a telephone) and the computer.
MPE	The Multiprogramming Executive operating system of the HP 3000 computer systems.
MPE defaults	Values for parameters or various system characteristics that take effect if a parameter is omitted from an MPE command or intrinsic.
$NEWPASS	A system-defined file name to be used when creating a special temporary file that can be referenced by a succeeding operation as $OLDPASS.
NOCCTL	A parameter of the :BUILD and :FILE commands that tells MPE that the program will not provide carriage control information.

$NULL	A system-defined file name that tells MPE to accept input or direct output to a dummy or "ghost" file, for example, output normally directed to a line printer can be discarded by writing it to a $NULL file.
offline state	The state of a device when it is not available to the system.
$OLDPASS	A system-defined file name to be used when referencing the temporary file most recently created with the name $NEWPASS.
opening a file	Initiating access to a file before beginning input/output operations with the file. Accomplished by calling the FOPEN intrinsic.
optional parameter	Parameter that is not required when entering a command or calling an intrinsic. In reference manuals, optional parameters are surrounded by brackets, [ ].
output priority	A number in the range of 1 (lowest priority) to 13 (highest priority) assigned to a file with the :FILE command. It is used to determine which spooled devicefile of all those waiting for a specific printer or card punch should be selected.
parameter	A variable assigned a constant value for a particular purpose or process, used in commands and procedure calls to request specific action.
PARM= parameter	A :RUN command parameter that passes a value to the program that can be used for control or other purposes.
password	A word, consisting of at most 8 alphanumeric characters beginning with an alphabetic character, assigned to an account, group, or user name when it is defined. A password must be supplied when accessing an account or group, or logging on with a user name for which a password is defined.
permanent file	A disc file that is saved and whose name is in the system directory until it is deliberately purged from the system.
physical record	In MPE terminology, a block of logical records.
process identification number (PIN)	A number assigned to a process by the operating system when the process is initiated. The number is used to communicate information about the process to the console operator.
production mode	The environment in which programs and jobs are being run to produce actual data rather than test data.
PUB group	A special group that exists in each account whose files are normally accessible in some way to all users within the account.
record	See logical record.
relative record	A number that represents the position of a logical record in relation to the first record in the file. The first record is numbered either 0 or 1 depending on the subsystem or utility being used.

run-time	The environment in which a program is running or executing.
save access to a group	The capability of making a file a permanent file on the system.
sector	A portion of a track on a disc device; the smallest addressable piece of the disc.
security	The provisions made by MPE to protect accounts, groups, files, and user names from unauthorized use.
sequential access	Reading from or writing to a device by accessing the records in consecutive order. This is the only type of access available with non-disc devices.
session	A mode of using the HP 3000 interactively, entering commands and data through a keyboard terminal and receiving immediate responses to your input. A single session begins when a :HELLO command is entered and ends when a :BYE command or another :HELLO command is entered.
session/job file directory	A directory maintained by MPE that contains the names of temporary files created during a session or job that are to be deleted when the session or job terminates.
session/job temporary file	A temporary file that is to be deleted when the session or job terminates.
SORT	The HP 3000 sort/merge program that allows you to sort records in a file into a prescribed order and to merge records by combining two or more previously sorted files into one sorted file.
spoofle	A spooled devicefile, a file read from or directed to a spooled non-sharable device, such as a card reader or line printer, and temporarily written on the disc.
spooled device	A device on which spooling has been enabled.
spooling	A facility to assist in the sharing of devices. The data coming from or going to a spooled device is copied by MPE to a disc and, in the case of output files, directed to the device when it is free. The records of spooled input devices are read in immediately, written to disc, and made available to the program as it requests them.
$STDIN	A system-defined file name that refers to the terminal on which a session is initiated or the device from which a job is initiated, (a card reader, tape, or terminal).
$STDINX	A system-defined file name for the session or job input device. When this file name is used, some MPE command images may be read as part of a data file.
$STDLIST	A system-defined file name that refers to the standard session or job listing device (usually the terminal on which a session is initiated and the line printer for a job).

stream, job	A set of job control commands and data delimited by a :JOB command record and an :EOJ command record, read into the system and scheduled for execution by using the :STREAM command.
subqueue priority	A category assigned to a process (an executing program) when it begins that determines how it will compete with other processes for access to the central processor. (See the *System Manager/System Supervisor Reference Manual* for a description of the available subqueues.)
subsystem	A term referring to software packages that perform specific functions such as compiling programs or editing text.
SYS account	A special account that is present on the system when it is delivered. It contains system files, the system Segmented Library, and supported subsystems such as the compilers.
system configuration	The process of modifying the operating system provided by the factory to suit the needs of your installation. (See the *System Manager/System Supervisor Reference Manual* for a description of this process.)
system console	A terminal through which an operator initializes and monitors the MPE operating system and requests various operations such as allocating devices, sending messages to user terminals, and shutting down the system.
system file directory	A directory maintained by MPE that records the name, group, and account of each permanent file on the system as well as other file information.
system manager	A person who manages the overall system by creating accounts and defining resource-use limits for each account.
system supervisor	A person who manages the system on a day-by-day basis, scheduling queues, altering the system configuration, maintaining the system library, and displaying various items of system information.
system-defined files	Files defined by MPE and made available to all users to indicate standard input or output devices.
system-file-name, COBOL	A parameter of the COBOL SELECT statement ASSIGN clause.
Systems Engineer	An HP technical software specialist.
temporary file	A file that exists only during the period in which a program has the file opened or the duration of a session or job.
undefined-length records	A record whose size varies and is only defined in terms of the maximum size allowed. Files with undefined records must have a blocking factor of 1 and the records are written without buffering.
user name	The formal name by which the system identifies a person and knows what capabilities the person is authorized to use on the system.

utility programs	Programs provided by MPE to perform specific functions for you. (See the *MPE Systems Utilities Reference Manual* for a description of these programs.)
variable-length records	Records that vary in size with respect to each other.
WHILE block	A set of three Editor commands beginning with a WHILE command, executed repeatedly.
word	16 bits or 2 bytes of information in the HP 3000 system.
word offset numbers	The number of words a location is removed from a base address.
work file, Editor	A file used by the Editor that contains the information being modified.

references

references

HP 3000 Operating Systems Except MPE-C

Part No.	Title
30000-90013	Console Operator's Guide
30000-90015	Error Messages and Recovery Manual
30000-90008	General Information Manual
30000-90009	MPE Commands Reference Manual
30000-90010	MPE Intrinsics Reference Manual
30000-90011	MPE Segmenter Reference Manual
30000-90044	MPE System Utilities Reference Manual
30000-90049	Software Pocket Guide
30000-90014	System Manager/System Supervisor Reference Manual
30000-90040	FORTRAN Reference Manual
30000-90079	KSAM/3000 Reference Manual
30000-90026	BASIC Interpreter Reference Manual
32215-90003	IMAGE Data Base Management System Reference Manual

HP 3000 Computer Systems Using MPE-C

Part No.	Title
32000-90004	Console Operator's Guide, 32000C MPE/3000
03000-90096	General Information Manual
32000-90002	MPE/3000 Reference Manual, 32000C
32000-90008	MPE/3000 Operating System, System Utilities Manual
03000-90126	Software Pocket Guide, HP 3000
32000-90006	System Manager/System Supervisor Manual, 32000C MPE/3000
32102-90001	FORTRAN/3000 Reference Manual
03000-90008	BASIC/3000 Interpreter Reference Manual
30000-90041	IMAGE Data Base Management System Reference Manual

All HP 3000 Operating Systems

Part No.	Title
03000-90121	Using the HP 3000: A Guide for the Terminal User
32213-90001	COBOL/3000 Reference Manual
32104-90001	RPG/3000 Compiler Reference and Application Manual
03000-90064	FCOPY Reference Manual
32214-90001	Sort/3000 Reference Manual
03000-90012	EDIT/3000 Reference Manual

index

index

READER COMMENT SHEET

USING FILES . . . a guide for new users of the HP 3000 computer systems

30000-90102 **April 1978**

We welcome your evaluation of this manual. Your comments and suggestions help us improve our publications. Please use additional pages if necessary.

Is this manual technically accurate?	**Yes** ☐	**No** ☐	(If no, explain under Comments, below.)
Are the concepts and wording easy to understand?	**Yes** ☐	**No** ☐	(If no, explain under Comments, below.)
Is the format of this manual convenient in size, arrangement, and readability?	**Yes** ☐	**No** ☐	(If no, explain or suggest improvements under Comments, below.)

Comments:

Cut along dotted line.

FOLD FOLD

FIRST CLASS
PERMIT NO. 1020
SANTA CLARA
CALIFORNIA

BUSINESS REPLY MAIL

No Postage Necessary if Mailed in the United States. Postage will be paid by

Publications Manager
Hewlett-Packard Company
General Systems Division
5303 Stevens Creek Boulevard
Santa Clara, California 95050

FOLD FOLD

FROM:

Name	______________________
Company	______________________
Address	______________________
	______________________
	______________________

Part No. 30000-90102
Printed in U.S.A. 4/78
3MPE.310.30000-90102

HEWLETT hp PACKARD